Trabajo Fin de Máster
Máster en Ingeniería Aeronáutica

Diseño, desarrollo y evaluación del software de vuelo de la misión CEPHEUS

Autor: Pablo Plaza Gomariz

Tutor: José Manuel Quero Reboul

Dep. Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2016

Trabajo Fin de Máster
Máster en Ingeniería Aeronáutica

Diseño, desarrollo y evaluación del software de vuelo de la misión CEPHEUS

Autor:
Pablo Plaza Gomariz

Tutor:
José Manuel Quero Reboul
Profesor Titular

Dep. Ingeniería Electrónica
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2016

Trabajo Fin de Máster: Diseño, desarrollo y evaluación del software de vuelo de la misión CEPHEUS

Autor: Pablo Plaza Gomariz
Tutor: José Manuel Quero Reboul

El tribunal nombrado para juzgar el trabajo arriba indicado, compuesto por los siguientes profesores:

Presidente:

Vocal/es:

Secretario:

acuerdan otorgarle la calificación de:

El Secretario del Tribunal

Fecha:

Agradecimientos

Para la realización de este trabajo ha sido fundamental el apoyo y ayuda ofrecidas por el profesor José Manuel Quero Reboul, así como por la empresa SolarMEMS. Durante todo el proceso han puesto a nuestra disposición el material, los espacios de trabajo, el apoyo y los consejos necesarios para su culminación.

Este trabajo materializa el final del periodo universitario, a lo largo del cual ha sido fundamental el apoyo recibido por mi familia. Gracias por el infinito tiempo y cariño que me habéis dedicado. Sin vosotros esto no habría sido posible.

Pablo Plaza Gomariz
Sevilla, 2016

Resumen

En el presente trabajo se recoge la base documental y el proceso inicial de elaboración de software para la plataforma cubesat del proyecto CEPHEUS.

El proyecto se distribuye en un Capítulo inicial sobre el estado del arte y las técnicas de programación empleadas, así como una presentación sobre el proyecto de construcción del satélite. A continuación se muestran los requerimientos iniciales de la misión y las características principales del proceso de planificación. En el Capítulo de Estructura del Proyecto se presenta la gestión de la documentación necesaria para llevar a cabo un proyecto de esta características, además de distintas herramientas útiles de gestión para el control de la calidad del proceso. A continuación, en el Capítulo Diseño se tienen los diagramas funcionales y características de interés de las principales tareas que debe llevar a cabo los sistemas de la plataforma.

Por último se han descrito las interfaces, las entradas y salidas y las definiciones de los hilos principales, así como las funciones que inicializan los principales sistemas. El proyecto finaliza con las pruebas funcionales realizadas para validar el diseño e implementación del código, seguido de las conclusiones y trabajos futuros que pueden ser de interés para la consecución de una plataforma plenamente funcional y operativa.

Abstract

In this paper, the documentary base and the initial software development process for a cubesat platform of the CEPHEUS project are collected.

The project is distributed in an initial Chapter based on the state of the art and programming techniques used, as well as a presentation on the satellite construction project. The initial requirements of the mission and the main features of the planning process are shown in next Chapter. The Project Structure Chapter presents the management of the documentation necessary to carry out a project of this nature, as well as various useful management tools for controlling the quality of the process. Next, in the design chapter we have the functional diagrams and characteristics of interest of the main tasks that the systems of the platform must carry out. Finally, interfaces, inputs and outputs and the definitions of the main threads, as well as the functions that initialize the main systems have been described. The project ends with functional tests to validate the design and implementation of the code, followed by conclusions and future work that may be of interest for the achievement of a fully functional and operational platform.

Índice Abreviado

Índice

1 Introducción

En 1903 el profesor Konstantín Tsiolkovski, padre de la cosmonáutica, publicó el primer tratado académico sobre el uso de cohetes para la exploración espacial. Con *La exploración del espacio cósmico por medio de los motores a reacción* se sentaron las bases para el desarrollo de misiones fuera de la atmósfera. Gracias a su aportación comenzó a vislumbrarse como una posibilidad real la exploración del espacio.

Con la llegada de la guerra fría entre Estados Unidos y la Unión Soviética se produjo un gran desarrollo tecnológico del sector espacial como una expresión de fuerza por parte de ambas potencias. La conquista del espacio fue el objetivo primordial de estos dos países durante la segunda mitad del siglo XX. Fue un 4 de octubre de 1957, desde el Cosmódromo de Baikonur, cuando la Unión Soviética hizo historia; lanzó el primer satélite artificial de la historia de la humanidad, el Sputnik.

El Sputnik era un satélite muy rudimentario, una masa de 83 kg con forma esférica y dos transmisores. Orbitó la Tierra emitiendo señales periódicas durante tres semanas, hasta que se desgastaron las baterías internas. Fue seguido de cerca por gran parte de la población mundial, siendo un duro golpe para los Estados Unidos, quien consiguió poner en órbita su primer satélite en 1960.

Como bien es sabido, la carrera espacial de la guerra fría terminó con el alunizaje, en 1969, de la cápsula del Apolo 11. Estados Unidos había ganado la carrera espacial, abriéndose un periodo de cooperación mundial que dura hasta nuestros días.

Desde el Sputnik 1, se han puesto en órbita numerosos tipos de satélites, cada vez más complejos y pesados. Para tener un orden de magnitud, Rusia y Estados Unidos han conseguido poner en órbita en el año 2012 más de 1500 cargas útiles[11]. En cuanto al tamaño, los satélites de comunicaciones y observación terrestre pueden tener masas superiores a los 1000 kg y disponer de los últimos sensores y equipamiento.

Durante esta tendencia a aumentar el peso y la complejidad, en 1999, la California Polytechnic State University (Cal Poly) desarrollaron las especificaciones del CubeSat para ayudar a universidades alrededor del mundo a realizar proyectos de ciencia espacial. Un CubeSat es un satélite de pequeño tamaño, volumen de un litro y masa inferior a 1,33 kg [9]. Con este nuevo concepto se abre un amplio abanico de posibilidades. Se reduce tanto la complejidad como los costes de desarrollo y lanzamiento. El espacio es ahora accesible para instituciones como universidades o empresas de menor tamaño, las cuales pueden desarrollar plataformas para distintos fines; prueba de sensores y equipos, vigilancia terrestre, estudios científicos, etc.

Otro punto a tener en cuenta es el uso de software libre y de una comunidad que en gran medida comparte sus conocimientos. Esto no solo impulsa a participar a nuevos desarrolladores, sino que enriquece y mejora los procesos de toda la comunidad.

2 Estado del Arte

2.1 Proyecto CEPHEUS

El proyecto CEPHEUS nace en 2013 con el objetivo de integrar e implementar un picosatélite íntegramente andaluz. Los principales socios participantes son empresas de desarrollo tecnológico como: Abengoa Hidrógeno, SolarMEMS, Inabensa, Idener, Alter Technology y Mesurex.

Este cubesat sigue el estándar CubeSat de 3 Unidades, Figura 2.1, constando de cuatro cargas útiles científicas. La principal de ellas es una pila de combustible diseñada por Abengoa-Hidrógeno, para la investigación de nuevas tecnologías de diseño con aplicación espacial. Además se añaden dos dispositivos más bajo estudio, un Star Tracker de bajo coste, diseñado por la empresa Solar Mems y un transceptor experimental de la empresa Mesurex. La cuarta y última carga será un sistema de control de actitud avanzado, desarrollado por Idener.

Junto con el segmento espacio, se implementa también una estación terrena para captar las señales emitidas por el satélite así como para enviar las señales de comando al segmento espacio controlando la misión en todo momento.

2.2 Sistemas en tiempo real

Un sistema en tiempo real es todo aquel sistema informático que interactúa con su entorno físico y responde a una serie de estímulos en un tiempo determinado[10]. Las respuestas deben estar acotadas en el tiempo para que el sistema pueda realizar su tarea de forma efectiva. En el caso de satélites o CubeSats esto es un requisito fundamental, por ejemplo, un retraso en la activación del sistema de calefacción de las baterías o un retraso en el tiempo de respuesta en una comunicación puede poner en riesgo toda la misión.

Para la implementación de las funcionalidades en la plataforma se ha hecho uso de FreeRTOS (Real Time Operating System), una serie de bibliotecas y librerías en código C que permiten realizar algoritmos según

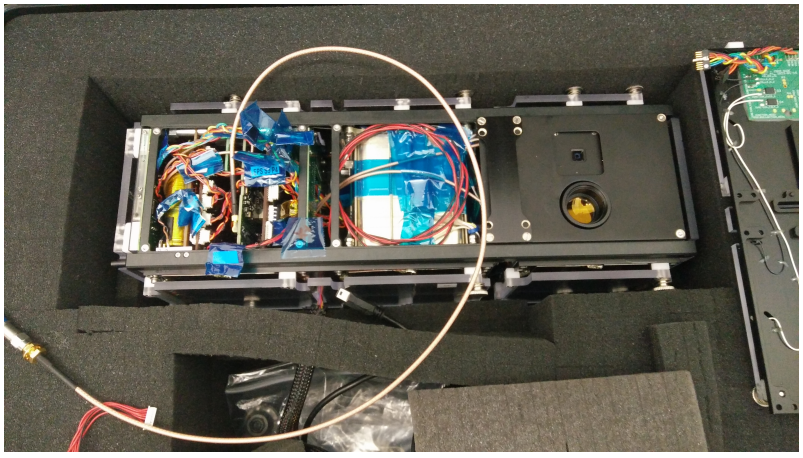

Figura 2.1 Cubesat CEPHEUS en su maletín portátil.

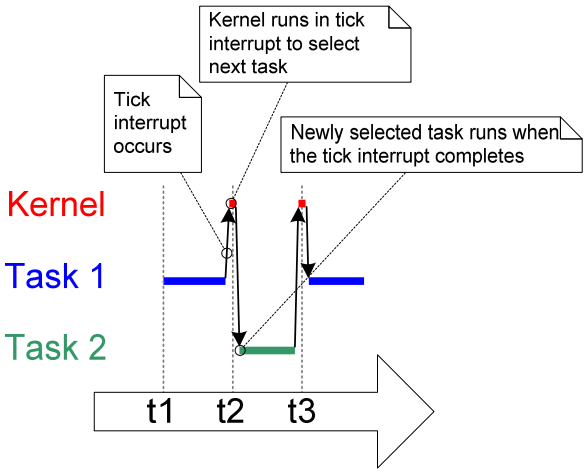

Figura 2.2 Ejemplo del proceso de multitarea [2].

la filosofía determinista de los sistemas en tiempo real [2]. Además permiten el desarrollo de multitareas en pequeños sistemas embebidos como el que dispone el picosatélite. Con este método de programación la plataforma llevará a cabo las tareas de gestión de potencia, cargas experimentales o comunicaciones de forma simultánea.

En la Figura 2.2 se puede observar el funcionamiento de la multitarea. La primera tarea se ejecuta durante un número determinado de ticks del sistema. Cuando se dispara la interrupción entra en funcionamiento el kernel que es la tarea que gestiona los demás hilos. El kernel identifica la siguiente tarea a realizar durante el próximo intervalo, hasta que se repite el proceso.

En estos tipos de sistemas se puede transferir información de un hilo a otro mediante el uso de colas. Gracias a estas estructuras se obtiene una elevada rapidez tanto en la transmisión como en la accesibilidad, ya que cualquier hilo puede enviar datos y recibir de cualquier otro.

En la Figura 2.3 se tiene un esquema del funcionamiento de las colas dentro del sistema freeRTOS. Cuando se envía a una cola un tipo de dato, este se coloca de forma predeterminada en el primer lugar, rellenando la cola hacia atrás. Cuando una tarea toma un dato de la cola este se elimina y todos pasan una posición hacia adelante, dejando un nuevo hueco libre.

2.3 Arquitectura de sistemas embarcados y filosofía de diseño KISS

En la Figura 2.4 se tiene un esquema de las funcionalidades que debe implementar todo sistema embarcado en un vehículo espacial.

El OBC contiene las siguientes funcionalidades:

- Detección de fallos, aislamiento y recuperación.

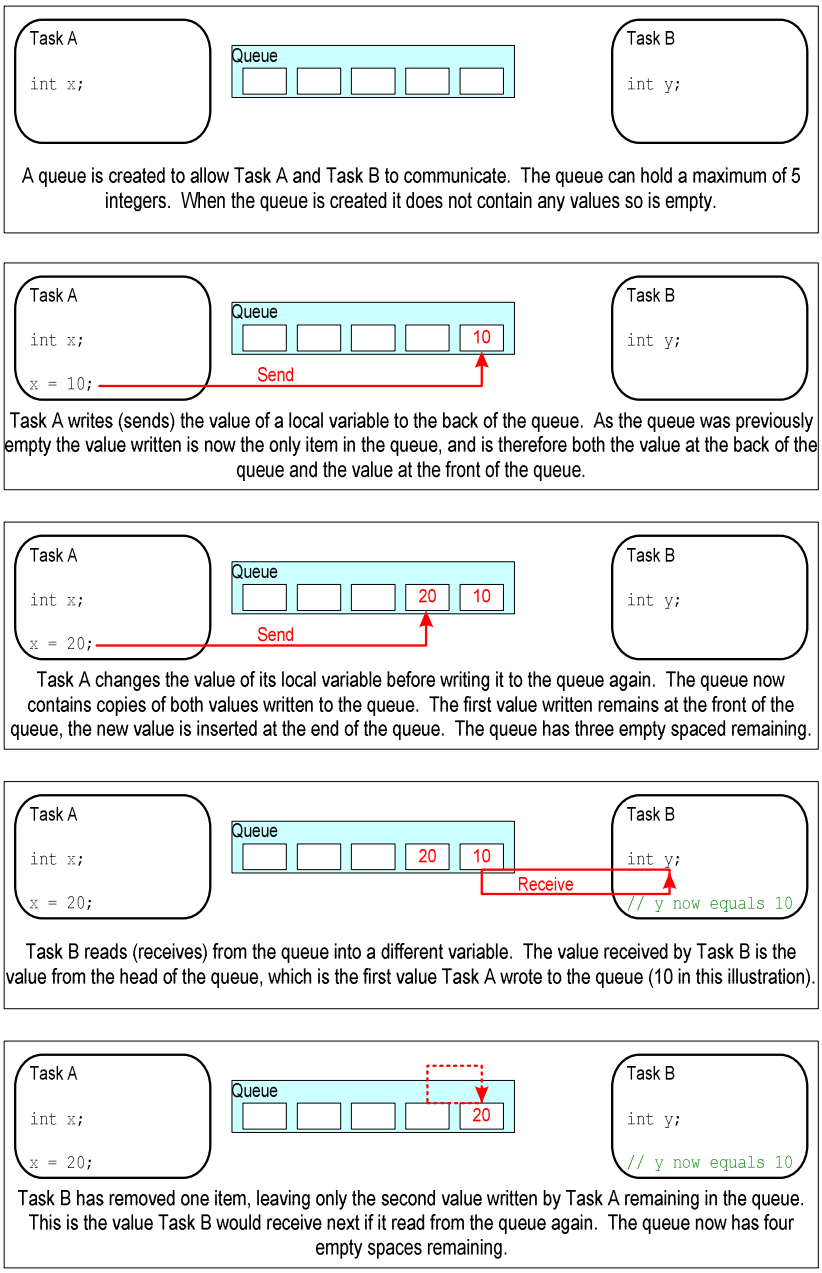

Figura 2.3 Esquema del funcionamiento de las colas en freeRTOS [2].

- Algoritmos de control
- Telecomandos y telemetría.
- Gestión de datos.
- Sistema operativo.
- Equipos embarcados.

La detección de fallos es uno de los grandes pilares sobre los que descansa el desarrollo de software espacial. Debido a las condiciones de operación de las plataformas no es factible que un error se propague dejando al sistema congelado. Un simple error que en tierra puede ser solucionado mediante el reseteo del equipo puede ser fatal en el entorno espacial. El sistema debe detectar los errores y gestionarlos, de forma que se garantice en todo momento la operatividad [8].

Para lo anterior se crea una herramienta fundamental, los llamados watchdogs (WDT), temporizadores que si no son refrescados en un intervalo de tiempo detienen la ejecución del código y resetean la plataforma.

Todo sistema de a bordo debe disponer de más de un (WDT) implementado como software o hardware; en el primer caso provocará lo que se conoce como un soft-reset y, en el segundo, un hard-reset.

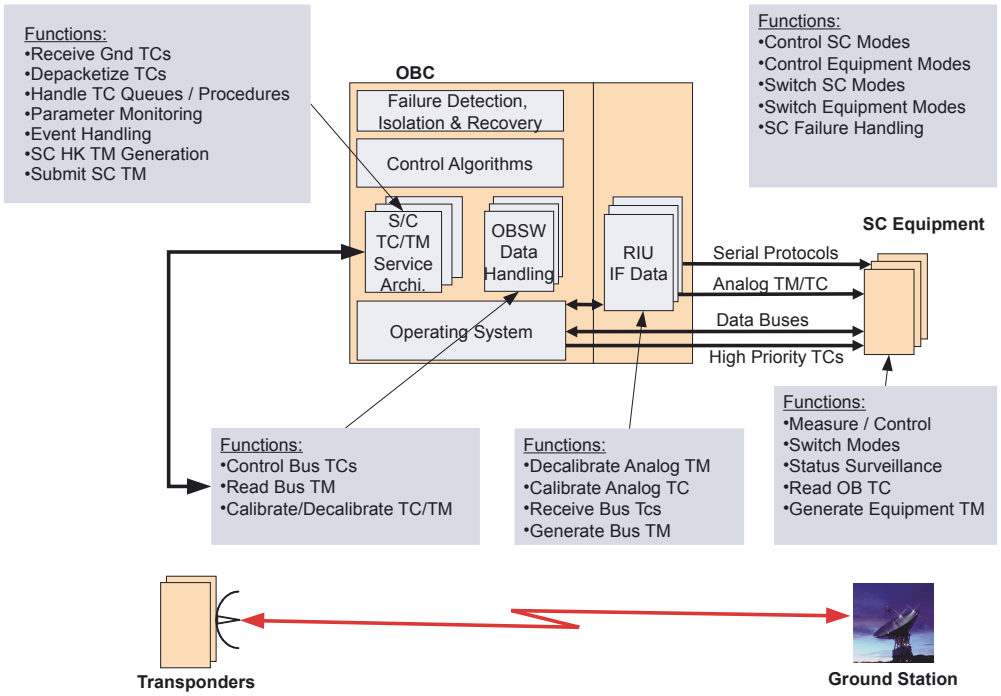

Figura 2.4 Esquema de las funciones de un sistema embarcado [20].

Otro aspecto a tener en cuenta es el aislamiento de fallos cuando estos no pueden ser resueltos. Estos casos suelen aparecer por la degradación de los componentes que están más expuestos a las condiciones extremas del entorno. Un ejemplo bastante conocido se da en las memorias no volátiles, las cuales son muy sensibles a la radiación y acaban por presentar defectos en su estructura interna que impiden la correcta escritura. El software debe estar diseñado para detectar este error y aislar este sector de memoria defectuoso.

La gestión de los telecomandos (TC) y la telemetría (TM) son críticos en el desempeño de la misión. Este proceso conlleva tanto la recepción y emisión como el desempaquetado, empaquetado y envío de la información completa al subsistema adecuado. Debe existir una identificación de hacia qué modulo va dirigido un telecomando así como una gestión del buffer de salida y de entrada, evitando que queden mensajes sin leer o enviar.

En general, cuando se trata de código destinado a productos espaciales es imperante el uso de la filosofía KISS (Keep It Simple Stupid). Como su propio nombre indica, siempre hay que intentar desarrollar el código de la manera más simple que sea posible. Con esto se obtiene el beneficio de que es fácilmente entendible y modificable por otro desarrollador, además de ganar en robustez ante fallos no contemplados. Para este proyecto, como no ha de ser de otra forma, se ha intentado seguir en todo momento este concepto, intentando realizar un código muy simple pero a la vez fiable.

3 Esquema de la misión

3.1 Planificación del proyecto

Para tener una visión general del proyecto se ha incluido las fases y tareas necesarias en el desarrollo de cualquier vehículo espacial, Tabla 3.1.

En el momento del desarrollo de este trabajo, el proyecto se encuentra dentro de las fases B/C y C/D, ya que la plataforma se encuentra físicamente preparada. Aún así, hay que tener en cuenta puntos de las fases iniciales, quedando el flujo de tareas para el desarrollo del SW de la siguiente forma:

- Definición de requisitos.
- Elaboración y estandarización de la documentación.
- Análisis de los requisitos de las cargas de pago.
- Diseño funcional de algoritmos.
- Diseño y definición de las interfaces.
- Desarrollo del software.
- Desarrollo de las pruebas funcionales.
- Verificación del cumplimiento de los requisitos.
- Desarrollo de los procedimientos operacionales.
- Comissioning, calibración y desarrollo de la misión.

En la Figura 3.1 puede observarse un diagrama de Gant de las distintas tareas a realizar a lo largo del desarrollo de un proyecto [15]. Para interpretar el esquema es necesario tener en cuenta que:

- PRR. Preliminary Requirements Review - Revisión de los requisitos preliminares de la misión. Se tiene en cuenta de forma general todos los aspectos de la misión que afectan al diseño.
- SRR. System Requirements Review - Revisión de los requisitos necesarios para los sistemas embarcados. Según el primer punto se disponen de unos requisitos que deben ser alcanzados mediante el uso de los subsistemas de la plataforma.
- PDR. Preliminary Design Review - Diseño preliminar. Se debe llegar a este punto con un esbozo de los diagramas funcionales de los sistemas de la plataforma.
- CDR. Critical Design Review - Identificación de los puntos críticos de la misión. Cierre del proceso de diseño de la plataforma.
- QR. Qualification Review - Punto final del proceso de desarrollo de mecanismos para la gestión de la calidad y la verificación de la plataforma.
- FAR. Flight Acceptance Review - Certificado de aceptación del producto para comenzar su vida operativa.

Tabla 3.1 Fases y tareas en el desarrollo de un proyecto espacial. Fuente ECSS-M30A..

Fase 0/A		Fase B/C & Fase C/D			Fase E
Evaluación de la misión y cumplimiento del diseño para las cargas de pago	Conceptualizar la misión, cargas de pago y diseño de la plataforma	Revisión y verificación del diseño		Producción, ensamblado, integración y test	Operaciones de la plataforma
1-Definición de los objetivos de la misión y las restricciones 2-Definición de la línea base y de las posibles variaciones 3-Análisis de los requisitos mínimos 4-Documentación	1-Análisis de los requisitos de las cargas de pago 2-Definición de cargas de pago alternativas 3-Análisis de las restricciones causadas por la órbita seleccionada 4-Estandarizar la documentación	1-Revisión y verificación del diseño 2-Desarrollo y verificación de las especificaciones de sistemas y equipos 3-Desarrollo y verificación del diseño funcional del algoritmo 4-Diseño de las interfaces	1-Subcontratación para la fabricación de componentes 2-Diseño detallado de componentes 3-Desarrollo y verificación de software 4-Desarrollo y v alidación de test 5-Test de subsistemas	1-Verificación de SW 2-Integración de sistemas 3-Validación del funcionamiento operacional 4-Desarrollo y verificación de los procedimientos de vuelo	1-Validación del segmento terreno 2-Entrenamiento de los operadores de vuelo 3-Lanzamiento 4-Inserción en la órbita 5-Calibración de las cargas 6-Evaluación del funcionamiento 7-Troubleshooting

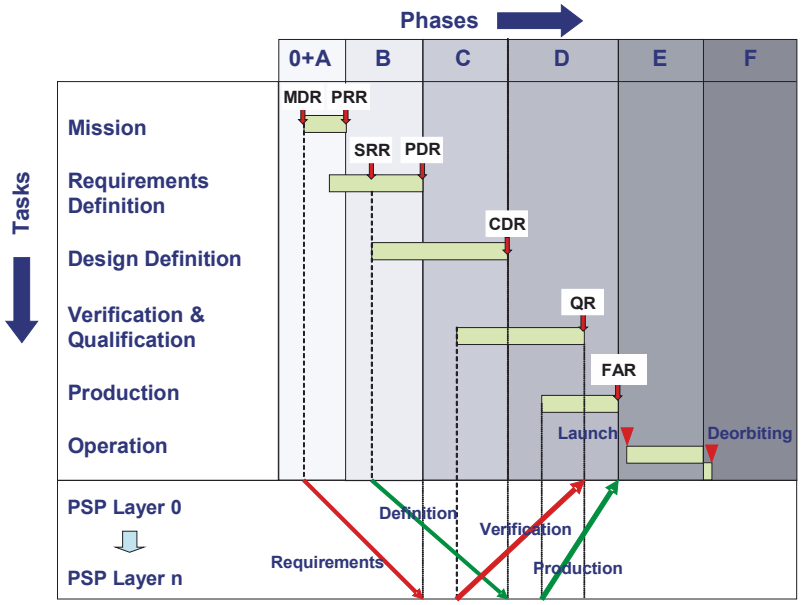

Figura 3.1 Fases y revisiones del desarrollo de una plataforma espacial. Fuente ECSS-M30A..

3.2 Requerimientos iniciales de la misión

En la definición del programa [19]se ha dispuesto que el satélite esté preparado para sobrevivir en un órbita de baja altitud (LEO) durante un periodo mínimo de dos años (RGDI-G-013). La órbita tendrá una excentricidad cercana a cero y se situará en torno a los 300-400 km de altitud (RGDI-O-019).

En este entorno se encuentra una presión de trabajo de 12 mbar (RGDI-G-018) y una temperatura de 67 ºC en la zona iluminada y -109 ºC en la zona de penumbra (RGDI-G-019). La plataforma debe estar diseñada para soportar estas condiciones extremas y seguir operando de forma continua sin poner en riesgo la misión. Es aquí donde se puede tener una idea del entorno tal hostil para los componentes que forman el satélite, especialmente los electrónicos. Es imperativo desarrollar un software robusto y eficaz, siguiendo la filosofía KISS, que tenga en cuenta estas restricciones y garantice en todo momento la supervivencia del satélite.

En esta sección se recogen los principales requerimientos de los tres sistemas principales del satélite: el sistema de gestión de datos (OBDH) y de control de actitud (ADCS), el sistema de comunicación y el sistema de gestión de las cargas de pago. Todos ellos han sido desarrollados en paralelo al tiempo que se realizaba el

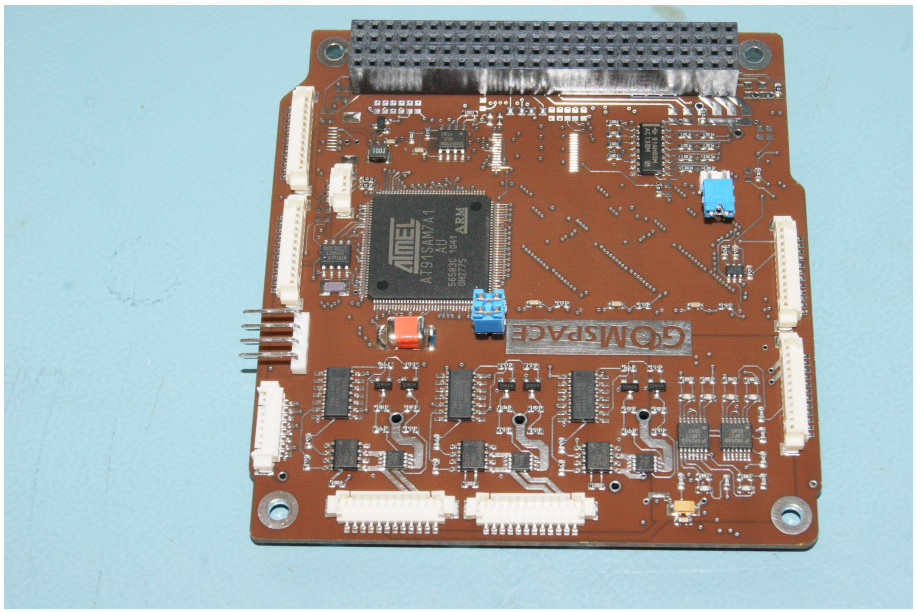

Figura 3.2 Placa Nanomind del sistema OBDH.

desarrollo de este proyecto, siendo los dos primeros los que son objeto de estudio en este trabajo.

3.2.1 Requerimientos iniciales del sistema OBDH

Los requerimientos que afectan al diseño del sistema OBDH se distribuyen en la tabla 3.2 y 3.3. Estos fueron establecidos en la fase inicial de todo el proyecto, por lo que para el trabajo que aquí se desarrolla algunos de ellos ya se han verificado. Es el caso de todos los requerimientos relacionados con el hardware del equipo para el OBDH, que han quedado fijados una vez que se optó por una plataforma de control Nanomind, Figura 3.2.

3.2.2 Requerimientos iniciales del sistema ADCS

Al igual que ocurre en el sistema OBDH, al comienzo del proyecto se introdujeron una serie de requerimientos que debían ser satisfechos por el sistema de control de actitud del satélite, como se recoge en la Tabla 3.4.

3.2.3 Requerimientos iniciales del sistema de comunicaciones

Aunque el sistema de comunicaciones no ha sido objeto principal de estudio de este trabajo, es necesario disponer de una visión global de las restricciones y requisitos, Tabla 3.5. Esto será de gran ayuda a la hora de diseñar un sistema de gestión de datos que sea acorde con los flujos de información entre el segmento espacio y el segmento terreno.

3.2.4 Requerimientos iniciales del sistema de cargas de pago

En cuanto al sistema de gestión de las cargas de pago, que son desarrolladas por los propios fabricantes, deben cumplir los requisitos fundamentales de la Tabla 3.6. En este proyecto se ha considerado unas cargas genéricas a la hora del desarrollo de la interfaz con la plataforma. Con esto se consigue un software modular que puede adaptarse a diferentes tipos de cargas[16].

Tabla 3.2 Requerimientos de tratamiento de datos.

ID	Descripción
RGDI-TD-001	Se debe garantizar la compatibilidad de todos los subsistemas y componentes con el bus de comunicaciones establecido por el centro de gestión del proyecto.
RGDI-TD-002	La adaptación de cada subsistema al protocolo global de comunicaciones del satélite será garantizada por su correspondiente desarrollador.
RGDI-TD-003	Las memorias no volátiles incluidas en el OBDH del satélite almacenarán los datos de telemetría y housekeeping del sistema completo y los códigos de software para las funciones de la plataforma y del ACDS avanzado.
RGDI-TD-004	Se podrá añadir alguna tarjeta SD de memoria adicional de hasta 2 GB en el ordenador de a bordo en caso de que el centro de gestión del proyecto así lo estime conveniente.
RGDI-TD-005	El almacenamiento de datos de las cargas científicas se gestionará por las mismas, incluyendo los dispositivos pertinentes dentro de su envolvente de potencia, volumen y masa.
RGDI-TD-006	Las cargas científicas transferirán datos al sistema OBDH de la plataforma exclusivamente cuando éste lo ordene, con el objetivo de transmitirlos a tierra.
RGDI-TD-007	Se reservará un mínimo de 2 MB de código en la memoria no volátil del OBDH para el software de la plataforma.
RGDI-TD-008	Se reservará un mínimo de 2 MB de datos en la memoria no volátil del OBDH para housekeeping y telemetría del satélite.
RGDI-TD-009	El microcontrolador del OBDH será de altas prestaciones y operará a 32-bits como mínimo.
RGDI-TD-010	La máxima cantidad total de telemetría y datos a transmitirdel satélite a tierra, en cada pasada coincidente con el segmento terreno, es de 6.9 Mbits.
RGDI-TD-011	La máxima cantidad total de comandos a transmitir de tierra al satélite, en cada pasada coincidente con el segmento terreno, es de 0.8 Mbits.

Tabla 3.3 Requerimientos operacionales.

ID	Descripción
RGDI-O-001	El satélite tendrá la capacidad de recibir y llevar a cabo un comando de apagado, de acuerdo con la reglamentación de la Comisión Federal de Comunicaciones (FCC).
RGDI-O-002	El tiempo de ejecución de la orden de apagado o encendido de los subsistemas debe ser $\leq$ 1 s.
RGDI-O-003	Todos los componentes desplegables, como son las antenas, deberán esperar un mínimo de 30 minutos para su despliegue después de que los interruptores de lanzamiento se activen tras la eyección del P-POD.
RGDI-O-004	Las 48 horas posteriores a la inserción del satélite en su órbita, fuera del lanzador, serán reservados para la carga de las baterías y la puesta en marcha de los subsistemas básicos de la plataforma.
RGDI-O-005	El satélite no establecerá comunicación con tierra de ningún tipo desde el momento de integración del mismo dentro del P-POD hasta mínimo 45 minutos después de su despliegue en órbita.
RGDI-O-006	Los transmisores de RadioFrecuencia embarcados en el satélite deberán esperar un mínimo de 30 minutos para comenzar a transmitir después de que los interruptores de lanzamiento se activen tras la eyección del P-POD.
RGDI-O-011	El satélite se encontrará siempre en alguno de los siguientes modos de funcionamiento: modo inicial (tras el despliegue del vehículo lanzador), modo nominal y modo fallo.
RGDI-O-012	El modo nominal de las cargas científicas se corresponderá al apagado de las mismas, de modo que se activarán exclusivamente para la ejecución de un experimento previa orden de la OBDH.
RGDI-O-013	El satélite será diseñado para un tiempo de adecuado funcionamiento en órbita superior a 24 meses.

Tabla 3.4 Requerimientos sistema ADCS.

ID	Descripción
RGDI-O-026	Los sensores de posición y actuación acomodados en el satélite serán exclusivamente sensores magnéticos y solares.
RGDI-O-027	Los únicos actuadores disponibles en el satélite serán de tipo magnético.
RGDI-O-029	Se garantizará un mínimo control sobre el spin del satélite con el control de actitud básico, procurando una baja velocidad de rotación durante su órbita alrededor de la tierra.
RGDI-O-030	El control de actitud básico buscará la alineación del eje z del satélite con el campo magnético terrestre.
RGDI-O-031	El error máximo en la actuación del satélite, garantizada por el control ADCS básico, será de un 10 %.
RGDI-O-032	El máximo momento magnético aportado por el sistema de control de actitud del satélite es de 0.2 Am^2.

Tabla 3.5 Requerimientos sistema de comunicaciones.

ID	Descripción
RGDI-I-013	El protocolo de comunicaciones del bus de datos será el protocolo I2C.
RGDI-I-016	La señal de reloj del ordenador de a bordo tendrá una velocidad fijada 40 MHz.
RGDI-I-018	La transferencia de datos a través del bus I2C se realiza de modo bi-direccional, a través de 2 cables, donde el único maestro es el OBDH de la plataforma.
RGDI-I-019	Todas las cargas científicas funcionarán como esclavos en la comunicación por el bus I2C.
RGDI-I-020	El tiempo de respuesta entre maestros y esclavos debe ser A,R siempre <1 ms.
RGDI-I-021	La velocidad nominal de transmisión de datos por el bus I2C A,R es de 337 kbit/s.
RGDI-I-022	El búfer hardware de transmisión es de 68 bytes.
RGDI-I-023	El búfer hardware de recepción es de 68 bytes.
RGDI-I-024	El tamaño máximo de los paquetes de datos transferidos por las cargas científicas a la OBDH será de 1 Kbits por cada experimento realizado.
RGDI-I-027	El Downlink entre el segmento terreno y espacio se realizará a una frecuencia de 425 MHz, en banda UHF.
RGDI-I-028	La capacidad máxima de transmisión en Downlink será de 9.6 Kbits/s.
RGDI-I-029	El Uplink entre el segmento terreno y espacio se realizará a una frecuencia de 144 MHz, en banda VHF.
RGDI-I-030	La capacidad máxima de transmisión en Uplink será de 1.2 Kbits/s.

Tabla 3.6 Requerimientos cargas de pago.

ID	Descripción
RGDI-SP-005	Los experimentos a llevar a cabo por cada carga serán diseñados por los desarrolladores de las mismas, debiendo posteriormente ser confirmados y autorizados por el centro de gestión del proyecto.
RGDI-SP-007	No se permitirá la ejecución simultánea de dos experimentos durante la misión en órbita.
RGDI-SP-008	No se permitirá la ejecución de dos experimentos durante un mismo período orbital.
RGDI-SP-009	No se permitirá la ejecución de ningún experimento durante el período en el que el satélite se comunique con el segmento terreno
RGDI-SP-010	No se permitirá la ejecución de ningún experimento hasta un mínimo de 48 horas después de la expulsión del satélite del P-POD y su puesta en órbita.
RGDI-SP-011	El número máximo de experimentos permitidos por cada 24 horas será de 5.
RGDI-SP-012	No se permitirá la ejecución de ningún experimento que requiera un aporte de energía superior a 2400 J.
RGDI-SP-013	Las cargas científicas se mantendrán en modo apagado de consumo mínimo siempre que no se encuentren ejecutando ningún experimento.
RGDI-SP-014	Se procurará siempre minimizar la potencia requerida por las cargas durante la misión, coordinando los experimentos y diseñándolos con dicho objetivo.
RGDI-SP-015	Todas las cargas científicas deberán incluir una interfaz adaptada para su comunicación con el resto de los subsistemas por el bus de potencia y el bus de datos I2C.
RGDI-SP-016	La interfaz de cada carga científica incluirá un relé de estado sólido que permita desconectar eléctricamente la carga científica del bus de potencia.
RGDI-SP-017	Todas las cargas científicas deberán incluir en sus datos el estado del subsistema correspondiente.
RGDI-SP-018	Todas las cargas científicas deberán incluir sus propios sensores de temperatura que mida su valor al menos cada 60 segundos con un error máximo de ±2°C.

4 Estructura del proyecto CEPHEUS

4.1 Plan de gestión del software

El software difiere de otras disciplinas y áreas de diseño del satélite en que no produce calor, ni tiene masa, ni posee otras características físicas que puedan ser comprobadas. La actividad de diseñar y desarrollar software es puramente intelectual y su salida es principalmente documental: si el software es considerado como documentación digital, su salida es también documentación.

4.1.1 Configuración del software

La configuración del software es un conjunto de actividades destinadas a gestionar el desarrollo y los cambios a lo largo del ciclo de vida del software. Para ello se define la línea base (baseline) de la configuración de la siguiente forma [13]:

- Fase I: Planificación de la gestión del sofware.
- Fase II: Recopilación de requisitos.
- Fase III: Fase de Diseño.
- Fase IV: Fase de Codificación.
- Fase V: Plan de verificación; test y pruebas.

4.1.2 Planificación

La planificación del software está referida a cómo se va a organizar el diseño y el desarrollo del software en los siguientes términos:

- Objetivo del software: relacionado con el objetivo de la misión.
- Requisitos: unidades de medida, requerimientos técnicos del software, funcionalidades, etc.
- Fase de diseño: requerimientos de diseño, reglas y normas, diagramas de flujo.
- Desarrollo: plataforma y equipo, códigos, etc.
- Tests y pruebas: tests de control de los requisitos, requerimientos, etc.
- Validación del alcance (baseline).
- Entrega.

4.1.3 Objetivos del software

Diseñar y desarrollar el software del ordenador de a bordo del satélite cubesat CEPHEUS, que será instalado en una plataforma NANOMIND A712D de la compañía Gomspace, y que irá embarcado en un satélite 3U con diferentes subsistemas y cargas útiles en su entorno.

4.1.4 Alcance y requisitos

El software deberá cumplir con las siguientes especificaciones:

- Instalación y operación en un ordenador NANOMIND A712D con sistema operativo FreeRTOS, de la compañía Gomspace.
- Realizar las siguientes tareas:
  - On Board Data Handling (OBDH):
    - Gestión de estado del satélite (housekeeping).
    - Gestión de la energía del satélite.
    - Organización de la memoria interna.
    - Gestión de fallos.
  - Communication system:
    - Gestión de las comunicaciones I2C con subsistemas y cargas.
    - Gestión de las comunicaciones con el segmento terreno.
  - Attitude and Orbit Control System (ADCS):
    - Control ADCS (actitud del satélite).
    - Muestreo de sensores y actuación del magneto-torque.
  - Payload On-Board Software:
    - Comunicación con las cargas.
    - Gestión de los experimentos.
    - Organización de los datos de cargas en memoria del ordenador de a bordo.
- Cumplir con los requerimientos de tratamiento de datos.
- Cumplir con los requerimientos de operaciones durante la misión.
- Cumplir con los requerimientos de sistema de ADCS básico.
- Cumplir con los requerimientos del sistema de comunicaciones.
- Cumplir con los requerimientos de cargas de pago.

4.1.5 Métodos y métricas

Para asegurar el cumplimiento de los requisitos se utilizarán las siguientes metodologías de trabajo y unidades de medida:

- Modularidad absoluta del software.
  - Diagramas de flujo de cada módulo y del sistema completo.
  - Código de versiones en cada módulo, con el formato vXX, comenzando en 00.
  - Cada módulo debe llevar un comentario con los siguientes datos, en la primera línea del código del archivo principal:
    - Nombre del módulo.
    - Descripción/Funcionalidad.
    - Inputs y outputs del módulo.
    - Autor/es.
    - Versión actual.
    - Fecha de creación.
    - Última fecha de modificación (versión).

        * Histórico de control: modificaciones realizadas en cada versión desde la primera hasta la actual.
- Idioma utilizado en los códigos: inglés.
- Comentarios en los códigos.
- Unidad mínima de memoria: byte.
- Unidad mínima de bit/rate: bit/segundo.

4.1.6 Diseño y reglas

Previo al desarrollo del código, se debe hacer el diseño de los módulos, acorde con las siguientes reglas:

- Regla 1. Diseñar el diagrama de flujo antes de escribir código.
- Regla 2. Los diagramas de flujo deben ser explícitos. No se debe omitir nada.
- Regla 3. Modular el software todo lo posible, para poder "reusar" el código según necesidad.
- Regla 4. Las variables más importantes deben estar accesibles para otros módulos, especialmente las de housekeeping.

4.1.7 Desarrollo del software

Para el desarrollo del código se deben seguir las siguientes reglas:

- Regla 1. Claridad del código antes que optimización del código. Cuánto más claro, más fácil de revisar, y por tanto más fácil de validar.
- Regla 2. No se deben ignorar los warnings.
- Regla 3. Cada función debe empezar con un comentario que describa su funcionalidad, sus inputs, y sus outputs. Si hay diferentes retornos, se deben indicar todos en dicho comentario.
- Regla 4. Cada archivo debe empezar con un comentario que describa su funcionalidad.
- Regla 5. Cada variable debe incluir un comentario en el momento de su definición/inicialización que describa su funcionalidad y unidades.
- Regla 6. Inicializar variables antes de usarlas.
- Regla 7. Los "define" deben estar en mayúsculas.
- Regla 8. Comprobar los "returns" en las funciones implementadas.
- Regla 9. Usar "enums" como tipos de error en retorno de funciones.
- Regla 10. Comprobar los "inputs" en las funciones implementadas.

4.1.8 Plan de verificación

La verificación del software desarrollado se realizará en dos niveles y siempre por el equipo de software:

- Verificación de requisitos que aparecen en el Capítulo 3.
- Verificación de la funcionalidad del código: se verificarán los diferentes módulos de la siguiente manera:
    - Cumplimiento de las reglas de diseño.
    - Cumplimiento de las reglas de desarrollo.
    - Coherencia y correspondencia entre los diagramas de flujo descritos y los códigos implementados.
    - Análisis exhaustivo de las funciones implementadas:
        * Comprobación de los inputs.
        * Comprobación de los outputs.
        * Comprobación de la funcionalidad.
        * Comprobación de los diferentes valores de retorno.

4.2 Plan de gestión de la documentación

Para la gestión de la documentación generada en el desarrollo e implementación del software del cubesat se ha elaborado un documento donde se recogen una serie de normas y directrices. Se ha empleado la plataforma de alojamiento de archivos Dropbox, estructurando las carpetas según se indica en la documentación.

4.2.1 Gestión de carpetas

Para tener un control completo de la documentación, se ha creado una carpeta por cada fase del proyecto:

- Requerimientos
  - Sistema de comunicación
  - Sistema OBDH
  - Sistema Pay-Loads
- Diseño preliminar
  - Sistema de comunicación
  - Sistema OBDH
  - Sistema Pay-Loads
- Diseño definitivo
  - Sistema de comunicación
  - Sistema OBDH
  - Sistema Pay-Loads
- Implementación
  - Sistema de comunicación
  - Sistema OBDH
  - Sistema Pay-Loads

En cada una de las sub-carpetas se guardarán los documentos pertinentes de cada sección, incluyendo una carpeta adicional con todas las versiones anteriores.

4.2.2 Identificación de los documentos

Cada documento seguirá un formato establecido en una plantilla dentro de la carpeta, además se nombrarán de la siguiente forma:

- Para los requerimientos: REQ-XX-VV, donde XX será OB, CO o PL según el departamento y VV la versión.
- Para el diseño preliminar: PRE-XX-VV
- Para el diseño definitivo: DEF-XX-VV
- Para la implementación: IM-XX-VV

4.2.3 Formato y estilo de los documentos

Para el formato de los documentos se ha tomado como referencia una plantilla la cual debe ser modificada con los identificadores del apartado anterior.

Es muy importante que cada versión nueva aparezca referenciada en la tabla de Historial del Documento, junto con una explicación clara y concisa de los cambios, la fecha y el autor. Además, todas las referencias a documentos deben estar reflejadas en la tabla Documentos de Referencia. Todos los documentos deben tener un índice con los contenidos, una lista de figuras y de tablas. Para realizar correctamente esto es necesario utilizar los comandos de títulos y de nombre de figura y tabla que implementa Word, ya que así se indexarán de forma automática y con un formato común. Los documentos presentarán una lista de abreviaturas, la cual será común para todos, y un objetivo del mismo.

4.2.4 Tareas y requisitos pendientes

En una carpeta Control, se tendrán tres documentos Excel; Cumplimiento Tareas, MOC y Planning. El primero servirá para rellenar todas las tareas pendientes de realizar. Se completará con el número de tarea, nombre, departamento solicitante, departamento responsable, fecha de apertura, fecha estimada de cierre y persona demandante. Cuando el responsable de cerrar este requisito lo lea, ha de rellenar la fecha de lectura y de finalización. La celda del extremo derecho aparecerá verde si falta más de una semana para cumplir la fecha estimada de realización, naranja si falta menos de una semana, roja si se ha cumplido el plazo de entrega y azul si está finalizada. El número de tarea estará acompañado de la fase del proyecto a la que corresponde; por ejemplo REQ-12, la tarea número doce de la primera fase del proyecto (Requerimientos).

En el archivo Planning aparecerá un diagrama de Gantt, Apéndice A.3, que proporciona una visión general del Proyecto. Se representan las semanas del proyecto y las cuatro fases, dentro de cada una los hilos de los departamentos. Para conocer el estado de cada departamento en una fase del proyecto se tomará como porcentaje el número de requerimientos finalizados entre el total del Cumplimiento de Tareas, Apéndice A.2. Es muy importante que cada tarea se introduzca en el archivo. Antes de cada reunión se actualizará el archivo Planning para conocer la evolución del trabajo.

Por último la Matrix of Compliace (MOC), Apéndice A.1, presentará en la columna inicial todos los requerimientos de la misión y en la primera fila los nombres de los códigos de software. Cuando un software cumpla un requisito se marcará la casilla con una X. Gracias a esto se sabrá si todos los requisitos se satisfacen con los distintos códigos.

5 Diseño

5.1 Arquitectura del dispositivo

El picosatélite constará de dos partes diferenciadas e independientes para garantizar el éxito de la misión con independencia de los resultados de los equipos bajo investigación. De este modo, se distingue una plataforma satelital autónoma que garantiza el tiempo de vida del dispositivo de acuerdo con el diseño y las cargas útiles compuestas por los experimentos a llevar a cabo, Figura 5.1.

La plataforma consta de un sistema de potencia para su funcionamiento (EPS) formado por paneles solares y baterías que suministran toda la potencia necesaria para el funcionamiento autónomo durante todo el tiempo de vida del sistema. Además, dispondrá de un básico sistema de control (ADCS) que procure un movimiento orbital adecuado para satisfacer el tiempo de vida de diseño junto con otro ADCS avanzado que probará novedosos sistemas de control optimizados sujetos a investigación. El sistema de telecomunicaciones (TMTC) llevará a cabo los enlaces de downlink y uplink entre la plataforma y el segmento terreno localizado en Sevilla. El control de todo el sistema, el procesamiento de operaciones y la gestión de los datos de todo el satélite se lleva a cabo por el sistema OBDH.

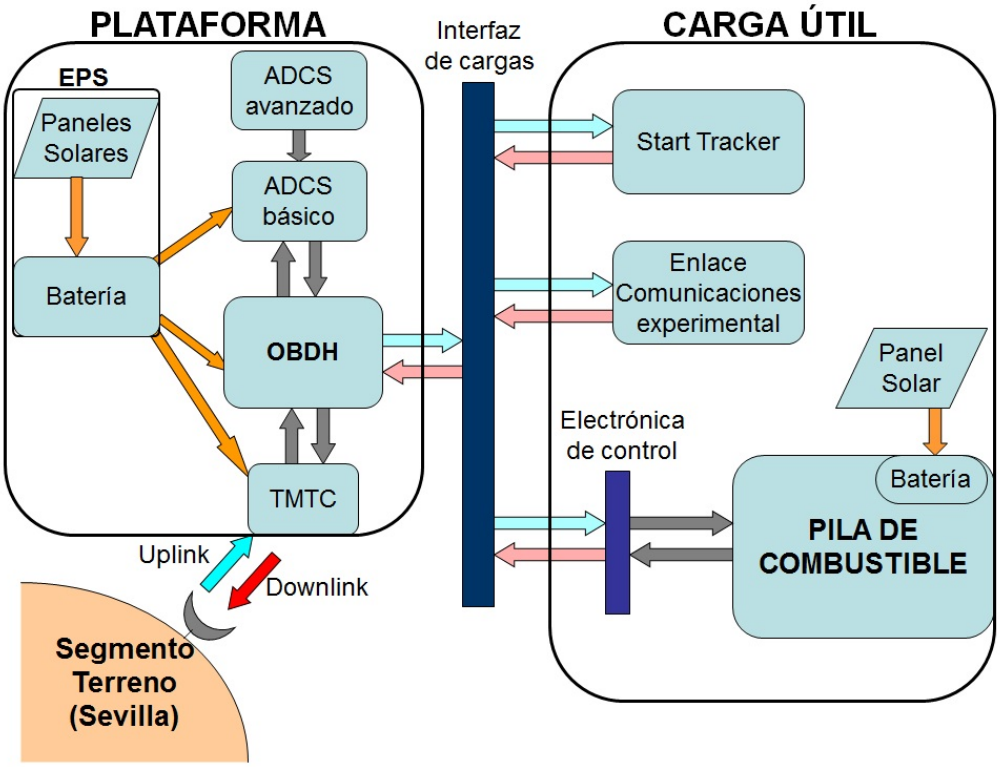

Figura 5.1 Arquitectura general del cubesat CEPHEUS.

La transferencia de datos entre la plataforma y las cargas útiles se realizará a través de una interfaz definida para tal fin, de modo que cada una de las cargas científicas deberá ser controlada y gestionada por sí mismas con los equipos que requiera para tal fin.

5.2 Funciones principales y prioridades del OBDH

El OBDH del cubesat debe cumplir todos los requisitos especificados en el documento y dar cobertura a las siguientes funciones por orden de prioridad:

1. Gestión de la energía del satélite
2. Gestión del housekeeping
3. Gestión de las comunicaciones
4. Gestión de errores
5. Gestión de los telecomandos
6. Gestión de memoria
7. Gestión de los experimentos

Se da la prioridad máxima a la energía, ya que ante todo es necesario mantener el satélite en funcionamiento el máximo tiempo posible. El housekeeping se encargará de recoger todos los datos del funcionamiento de los subsistemas y del log del software, para tener información de los posibles errores que puedan aparecer.

5.3 Máquina de estados

La operación del satélite depende de una máquina de estados con dos modos de funcionamiento:

- Modo nominal
- Modo supervivencia

En el modo nominal el satélite se encuentra completamente funcional, mientras que en el caso del modo supervivencia o safe mode se limitan ciertos aspectos. En este estado los experimentos quedan suspendidos y además el sistema de comunicación es avisado de este acontecimiento, con lo cual modela su comportamiento para adecuarse a las nuevas condiciones.

El modelado de esta máquina de estados está implementado en el hilo de gestión de potencia, el cual emplea el HK proveniente de las baterías para determinar el estado de las mismas, Figura 5.2. Si el voltaje, la intensidad o la temperatura no está dentro de los valores nominales el satélite entrará en modo supervivencia. Igualmente ocurre si aparece un fallo en el Real Time Clock (RTC).

5.4 Hilo y colas implementados

En la Figura 5.3 se muestra un esquema funcional de las tareas desempeñadas por el satélite y su relación entre ellas. La comunicación entre los distintos bloques, los cuales estarán formados por hilos, se realiza mediante el uso de colas. Nótese que en ningún momento se produce contacto entre el sistema de comunicaciones y las cargas de pago.

La Figura 5.3 es una simplificación de los bloques e hilos que componen el software del satélite. En la Tabla 5.1 se presentan todos ellos, junto con una breve descripción y la prioridad de la tarea. Estos hilos están relacionados mediante las colas de la Tabla 5.2.

5.5 Gestión de la energía

Este paquete de funciones es crítico para el satélite, en él se comprobará el estado del sistema de energía para determinar en qué modo de funcionamiento se encuentra CEPHEUS. Se tendrá en cuenta la tasa de carga-descarga, la intensidad, voltaje y la energía disponible para cambiar la variable de estado de acuerdo con las capacidades del satélite [5].

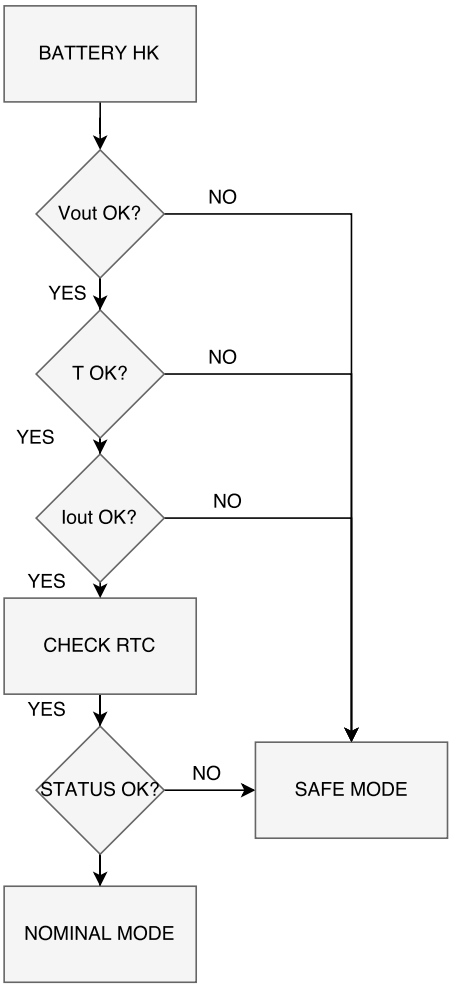

Figura 5.2 Diagrama máquina de estados.

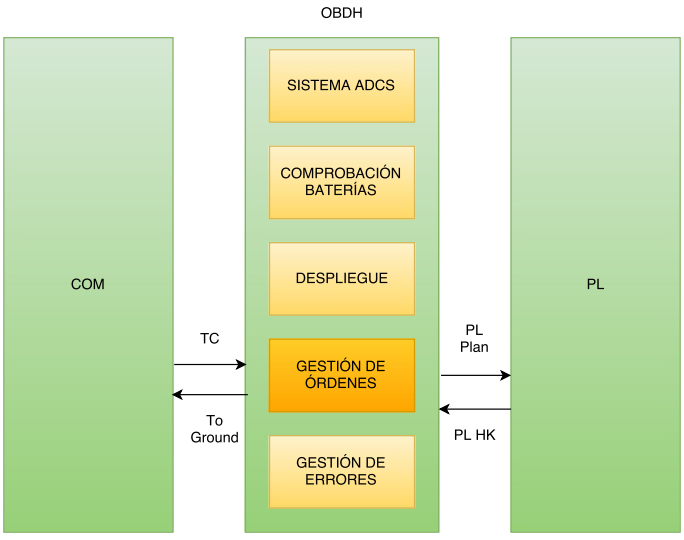

Figura 5.3 Esquema funcional CEPHEUS.

5.5.1 Nanopower

El cubesat emplea como sistema de gestión de la energía (EPS) una placa Nanopower, del fabricante Gomspace, Figura 5.4. Este dispositivo presenta unas baterías de ion-litio y está diseñado para suministrar energía a

Tabla 5.1 Configuración de hilos.

Descripción	Detalles	Prioridad	Nombre
Sistema ADCS	Detumbing y comprobación de la actitud del satélite, corrigiéndola si es posible	Baja	vTaskADCS
Comprobación baterías	Comprobar estado de las baterías, cambiando el estado del satélite	Máxima	vTaskPower
Despliegue	Despliegue de las antenas	Máxima	vTaskDeployment
Gestión de órdenes	Clasificación y ejecución de los telecomandos	Media	vTaskTC
Gestión de errores	Chequeo de todos los subsistemas y elaboración de log	Media	vTaskError
Gestión comunicación	Gestión del envío y recepción de datos entre el segmento espacial y terreno	Media	vTaskComunication
Tratamiento de datos	Preparación de paquetes para enviar y lectura de los recibidos	Media	vTaskComunicationData

Tabla 5.2 Configuración de las colas.

Origen	Destino	Descripción	Detalle	Nombre
OBDH	COM	HK y experimentos	Todos los datos que se quieran enviar se verterán a esta cola	xQueueToGround
COM	OBDH	Telecomandos	Todos los datos que sean recibidos desde tierra y no tengan como destino el módulo de comunicación	xQueueFromGround
OBDH	PL	Planificación de experimentos	Cola para disponer de la información y permisos al PL para la ejecución de experimentos	xQueuePlanner

pequeños satélites, con demandas que van desde 1 a 30 W. Presenta conexiones para las placas solares fotovoltaicas y convertidores de alta eficiencia para establecer el valor adecuado de tensión para cargar las baterías.

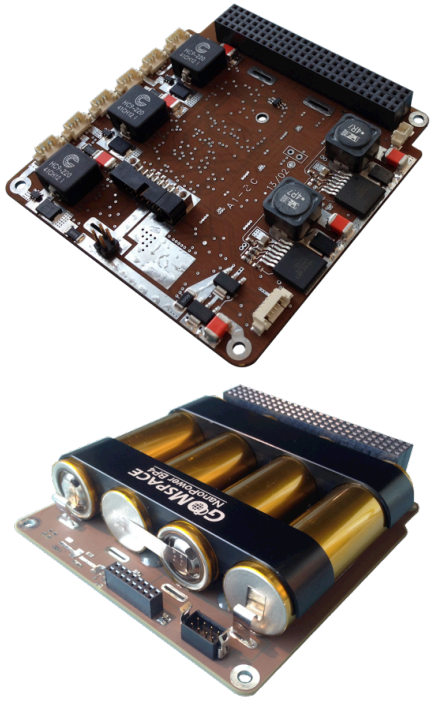

Figura 5.4 Dispositivo Nanopower.

La energía proveniente de las baterías es usada para alimentar dos convertidores, que alimentan con 3,3 V y 5 A o 5 V y 4 A a las cargas de toda la plataforma. En la Figura 5.5 se tiene diagrama de bloques con los componentes de una plataforma Nanopower [7].

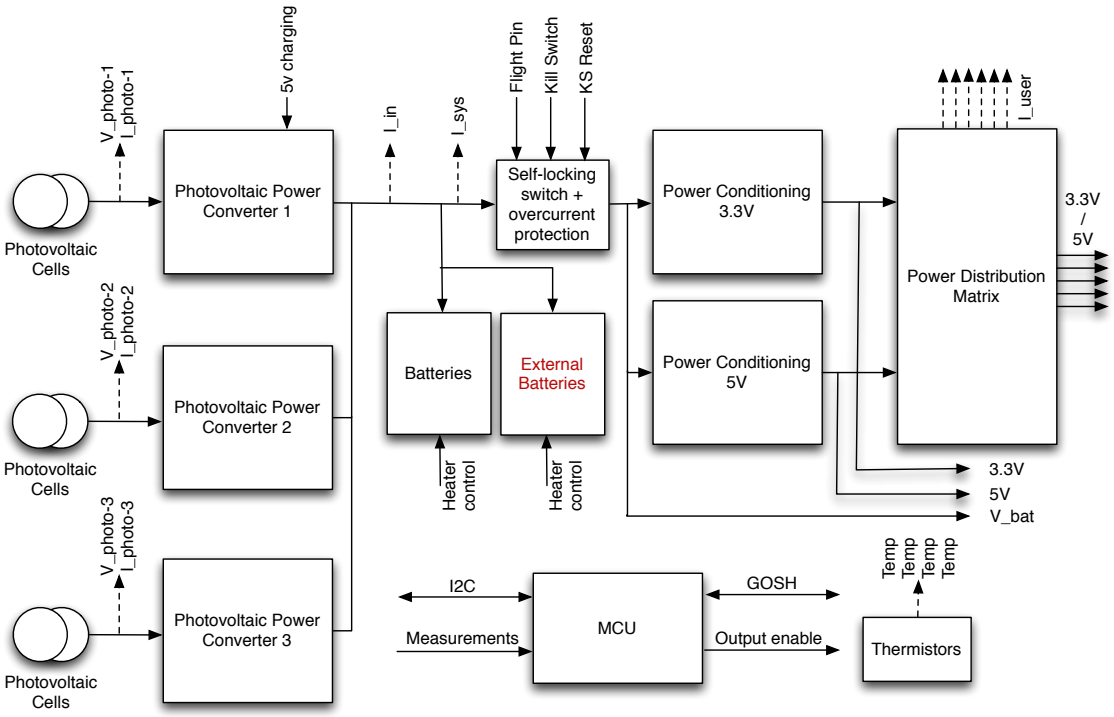

Figura 5.5 Diagrama de bloques de un dispositivo Nanopower.

Mode	Software level set points	Voltage level (V)
FULL (4)	Vmax	8.3 (*16.6)
NORMAL (3)	Vnormal	7.4 (*14.8)
SAFE (2)	Vsafe	7.2 (*14.4)
CRITICAL (1)	Vcritical	6.5 (*13.2)

** Only on Pxx-S*

Figura 5.6 Clasificación de voltajes.

Parameter	Condition	Min	Typ	Max	Unit
Lithium-Ion Cell					
- Voltage		3.0	3.7	4.2	V
- Charge current			1000	2500	mA
- Discharge current			1000	3750	mA
- Charge temperature		-5		45	ºC
- Discharge temperature		-20		60	ºC
- Storage temperature	80% recovery after 1 year	-20		20	ºC
- Internal impedance				70	mOhm
- Cycle life (20% capacity loss)	DOD: 100%, Temp 25degC Charge/discharge: 1C/1C		350		cycles

Figura 5.7 Datos técnicos de las baterías.

El módulo Nanopower P31us presenta una protección mediante hardware, según la cual por debajo de los 12,0 V se produce el apagado de todos los sistemas del satélite excepto el propio Nanopower. Esto afecta a todas las salidas de la placa, incluso las que se encuentran en pines de "siempre encendido". Cuando se esto se produce, el Nanopower solo realiza la carga de las baterías con los paneles solares, hasta que se llegue a un voltaje de 12,8 V. Por la parte de software, Gomspace presenta diferentes tipos de posiciones de voltaje, Figura 5.6.

Se ha estipulado por lo tanto que el voltaje mínimo para el funcionamiento nominal del satélite es 13,2 V. Si disminuye por debajo de este valor, el satélite entrará en modo supervivencia, eliminando las tareas que no sean fundamentales para la plataforma.

El módulo Nanopower dispone de un calentador específico para las baterías. Este puede ser encendido o apagado mediante comandos CSP desde el Nanomind, o bien configurados para que actúen de forma autónoma. Los calentadores son resistencias de 20 o 40 Ohmnios para cada dos baterías.

Según el requerimiento RGDI-G-019, y teniendo en cuenta las especificaciones de las baterías ofrecidas por Gomspace en su datasheet [6], Figura 5.7. Se ha optado por definir un funcionamiento nominal entre -20 y 60ºC, a partir de los cuales el satélite entraría en modo de supervivencia para garantizar la salud de las células.

Las baterías del Nanopower están protegidas contra cortocircuitos. El circuito monitoriza la intensidad del sistema, I_{sys}, y si esta sobrepasa el umbral un interruptor es abierto durante 100 ms. Esto cortará el suministro de todos los sistemas, tanto internos como externos. No tiene tiempo de recuperación, por lo tanto si se vuelve a sobrepasar una vez cerrado el circuito este se volverá a abrir durante 100 ms.

Se incorporará una función que controle la intensidad que se está demandando a las baterías, para evitar que se ejecuten más tareas que las soportadas por el sistema de potencia. Según la Figura 5.8 la intensidad máxima ofrecida por las baterías es 6,8 A.

5.5.2 Placas fotovoltaicas

Para la carga de las baterías se dispone de una placa fotovoltaica en cada cara del satélite. Los dispositivos han sido fabricados por la empresa ISIS y disponen de las siguientes características:

- Diodos en serie para prevenir que la corriente entre en las células solares. Cada célula dispone de diodos de protección para evitar que por el efecto de las zonas de penumbra se produzca la entrada de corriente de células conectadas en serie.
- Cada célula está diseñada para un mínimo de 5 años de vida en entorno espacial.

Parameter	Condition	Min	Typ	Max	Unit
Battery	Battery connection				
- Voltage		6.0 (*12.0)	7.40 (*14.9)	8.40 (*16.80)	V V
- Current, charge	(Depends on battery configuration)			6.00	A
- Current, discharge	Overcurrent protection threshold ***		6.8***		A
PV inputs	Photo-voltaic inputs				
- Voltage	(Customer selectable)	0 (*0)	4.2 (*8.4)	8.5 (*17)	V V
- Current, charge		0.00		2.00	A
5V_in	Battery charge input (beware of inrush)				
- Voltage	5 V => 0.9 A charge, 4 V => 0 A charge	4.10	5.00	5.00	V
- Current, cont.	@5 V		0.9	1.1	A
OUT-1,2,3,4,5,6	Latch-up protected outputs				
- Voltage	Configurable		3.3/4.98		V
- Current limit	Current cut-off limit (Cust. select)	0.5	Select	3.0	A
+5 V	5 V regulated output (always on)				
- Voltage		4.89	4.98	5.05	V
- Current, cont. **	Total current including output channels	0.005		4.00	A
+3.3 V	3.3 V regulated output (always on)				
- Voltage	From -40 to +85 °C	3.29	3.34	3.39	V
- Current, cont.	Total current including output channels	0		5.00	A
V_BAT	Raw battery voltage				
- Voltage	(Depends on battery configuration)	6.0 (*12.15)		8.40 (*16.80)	V
- Current out			12		A
Power consumption	Power consumed by NanoPower		115 (*210)		mW
Off current	Current consumed with separation switch OFF		35	60	uA
Shelf-life	Period until batteries are fully discharged when separation switch is OFF. *(Depends on battery configuration)*	700	1400		Days

* Only on Pxx-S
** A completely unloaded 5 V channel may show oscillations.
*** For higher threshold, please enquire at info@GomSpace.com.

Figura 5.8 Datos técnicos EPS.

- Presentan una eficiencia del 28 % al comienzo de su vida operativa.
- Disponen de un fotodiodo que actúa como sensor que mide la luz incidente.
- Disponen de un sensor de temperatura.

Toda la información se encuentra disponible en el datasheet realizado por ISIS, Anexo I.

5.5.3 Diagrama funcional

Para el correcto desarrollo de la misión, se ha elaborado un sistema de gestión de la energía sencillo pero a la vez robusto. Como se puede ver en la Figura 5.9, cada cinco minutos, o el intervalo fijado por el usuario, se hace una petición al EPS de housekeeping. Con los valores recibidos se estudia el nivel de carga de la batería, la temperatura y la intensidad. Si los dos primeros están fuera de rango el satélite entrará en modo supervivencia

5.6 Gestión del housekeeping

En este paquete se monitorizarán todas las variables posibles de los sistemas críticos del satélite para guardarlas en memoria y enviarlas cuando sea posible a la estación terrena. También se guardarán registros del software para conocer el correcto funcionamiento del mismo.

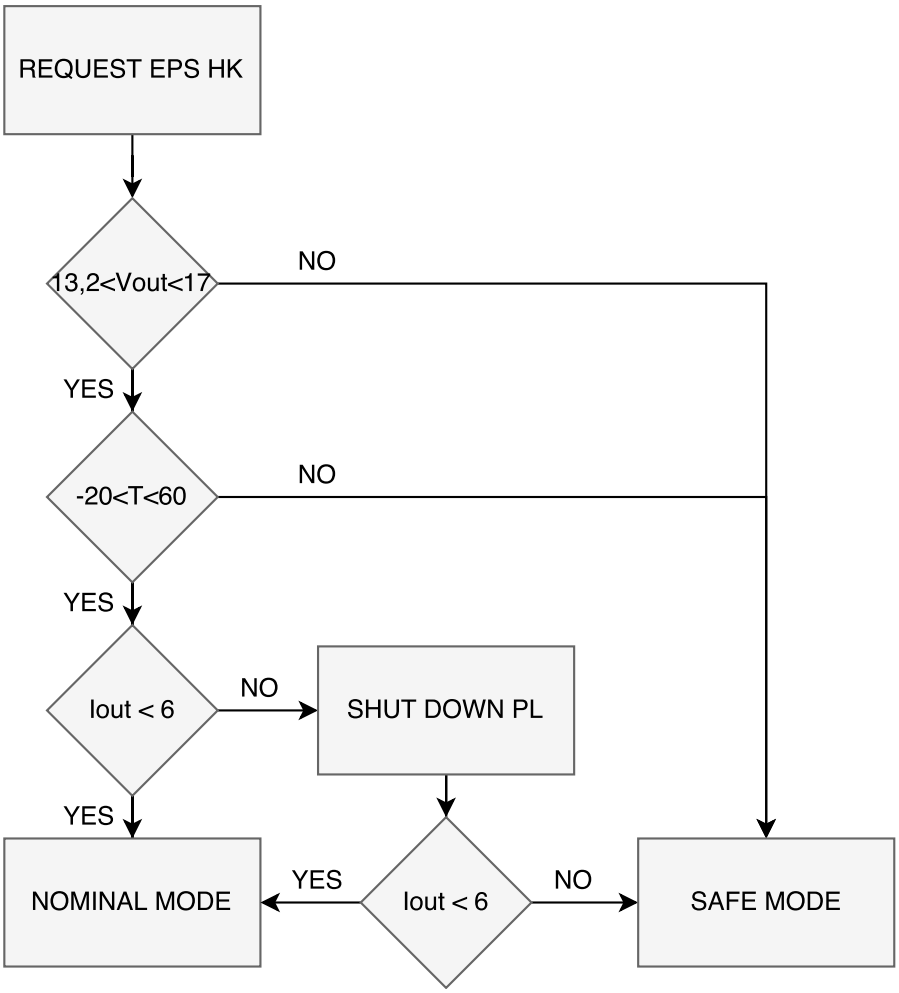

Figura 5.9 Diagrama funcional del sistema de gestión de potencia.

5.6.1 Diagrama funcional: HK solicitado desde el segmento terreno

El HK puede elaborarse mediante una orden recibida desde el segmento terreno. Este mensaje será desempaquetado por el sistema de gestión de datos y presentado en la cola xQueueFromGround. El hilo de gestión de telecomandos leerá esta entrada y preparará el HK, Figura 5.10

5.6.2 Diagrama funcional: HK en el modo baliza

Cuando el sistema de comunicación cambia su estado a modo baliza, el OBDH realiza periódicamente un HK completo de los subsistemas. Con esto se consigue que en la propia señal baliza se tenga una información completa del estado del satélite, sin necesidad de enviar una petición específica. En la Figura 5.11 puede observarse el proceso de petición a los distintos subsistemas del HK y su posterior escritura en la cola xQueueToGround.

5.7 Gestión de las comunicaciones

En este bloque se incluye tanto la interfaz entre el OBDH y el sistema de comunicaciones como la gestión de la memoria para situar los paquetes de envío en las posiciones correctas. También incluye la gestión de las órdenes desde el segmento terreno, tanto para el OBDH como para las cargas experimentales.

5.7.1 Diagrama funcional del sistema de comunicación

El sistema de comunicaciones dispone de tres estados: modo espera (wait), modo baliza (beacon) y modo conexión (connect), tal y como se puede ver en la Figura 5.12. El satélite se encuentra con el sistema de

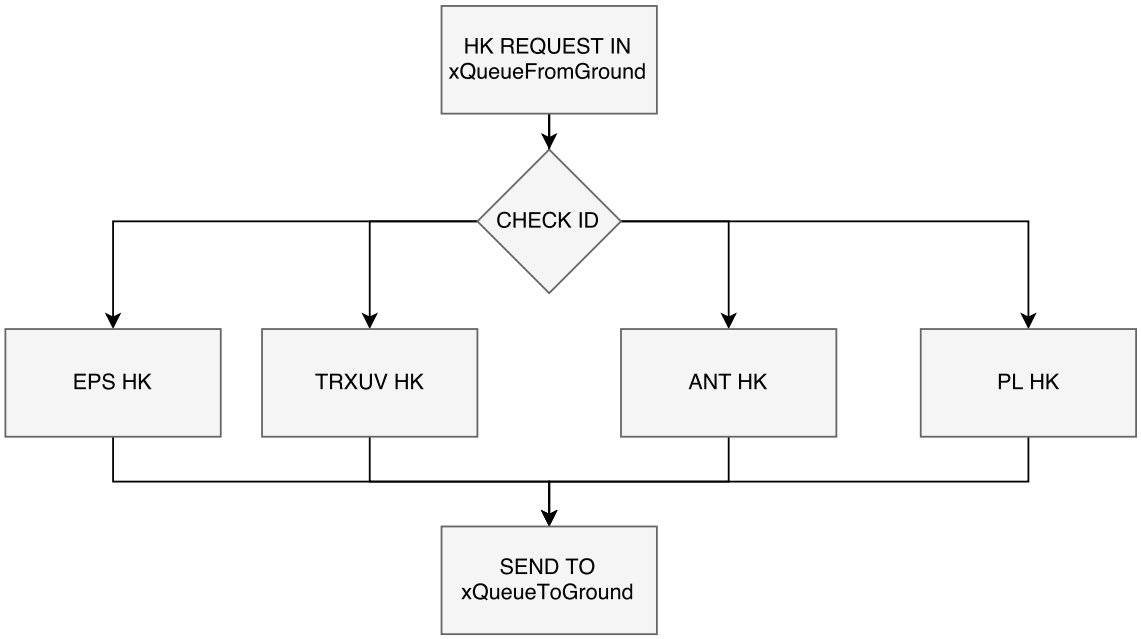

Figura 5.10 Diagrama funcional para solicitud de HK.

comunicación en modo de espera hasta que al consultar su RTC advierte de que una pasada por la estación terrena está cerca. En ese momento el sistema pasa a modo baliza, esperando la confirmación de conexión. Cuando esta es establecida puede comenzar la recepción y envío de datos, puede establecerse la conexión con un comando de autorización o de forzar conexión desde tierra. Cuando se recibe el telecomando de desconexión o expira el temporizador el satélite finaliza el estado de conexión y vuelve al modo de espera.

En el estado de espera, Figura 5.13, el sistema de comunicaciones envía una petición de información del estado del satélite al OBDH. Si se encuentra en el modo supervivencia o de despliegue se prepara para activar el modo beacon. Lo mismo ocurre si se acerca una hora de conexión y el satélite se encuentra en modo nominal. Si no es la hora de conexión y esta no ha pasado aún, se sigue con el modo en espera. Si se ha pasado se avisa al sistema OBDH para que actualice un nuevo estado en el satélite.

Para el modo baliza, Figura 5.14, se envía una solicitud de conexión, que si es aceptada, provoca que el sistema de comunicación establezca conexión. Dependiendo de si el satélite pasa de estado normal a supervivencia o viceversa y de la hora de conexión, el sistema regulará los temporizadores, avisara al sistema OBDH y entrará en modo espera si es necesario.

En el modo conexión, Figura 5.15, el sistema de comunicación comprueba que la autentificación es correcta y si dispone de los permisos del OBDH. Si esto es así, puede comenzar el envío y recepción de datos. Si no se dispone del permiso y no se fuerza la conexión, volverá al estado baliza. Por el contrario, si se fuerza la conexión desde tierra, se podrán enviar y recibir datos, aunque esto ponga en riesgo la integridad de la plataforma.

5.7.2 Despliegue de las antenas

El despliegue de las antenas se realizará, tal y como viene dispuesto en los requerimientos, 45 minutos después de la separación del P-POD. Esta acción es crítica, un fallo en el despliegue puede producir que el satélite no establezca conexión con el segmento terreno y ponga en peligro toda la misión.

El cubesat CEPHEUS dispone de 4 antenas, Figura 5.16, dos receptoras y dos emisoras, situadas en la parte superior. Se encuentran enrolladas y fijas mediante un hilo sensible al paso de la corriente eléctrica. Cuando se inicia el proceso de despliegue, el OBDH manda una corriente eléctrica a través del hilo, que se calentará y romperá, liberando el mecanismo de resorte de las antenas.

Hay que tener en cuenta que este tipo de operación requiere una elevada intensidad de corriente circulando por la plataforma, por lo que no es recomendable que se liberen los dos pares a la vez. Primero se liberarán dos y cuando se haya completado el proceso, si la carga de las baterías es suficiente, se liberarán las otras dos.

El sistema de despliegue de antenas dispone de sensores de temperatura y de comprobación de despliegue. Gracias a estos, tal y como se observa en la Figura 5.17, se ha optado por realizar una comprobación del

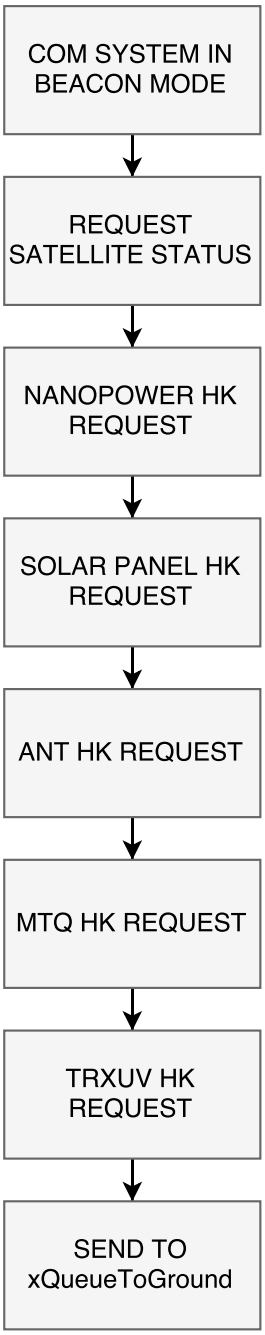

Figura 5.11 Diagrama funcional para solicitud de HK.

estado de las antenas cada vez que se reinicia el sistema, y si estas no se han desplegado se inicia el proceso anterior. Con esto se consigue un código sencillo pero a la vez muy robusto y resistente a errores.

5.8 Gestión de errores

La gestión de los errores es una parte fundamental de cualquier plataforma espacial. El satélite CEPHEUS incorpora una serie de tareas destinadas al seguimiento y comunicación de errores. En cuanto a la resolución, se disponen de varios WDT; dos implementados mediante software y un tercero mediante hardware.

Los "soft WDT" se encuentran situados en el EPS y otro en la plataforma OBDH. En el primer caso, la placa Nanopower reseteará las conexiones a los puertos de 3,3 y 5 V que están controlados por estos dispositivos y que no reciben una señal en un tiempo configurable por el usuario. En el segundo, se ha activado dentro del sistema freeRTOS de Nanomind un WDT que resetea la placa si uno de los hilos no deja

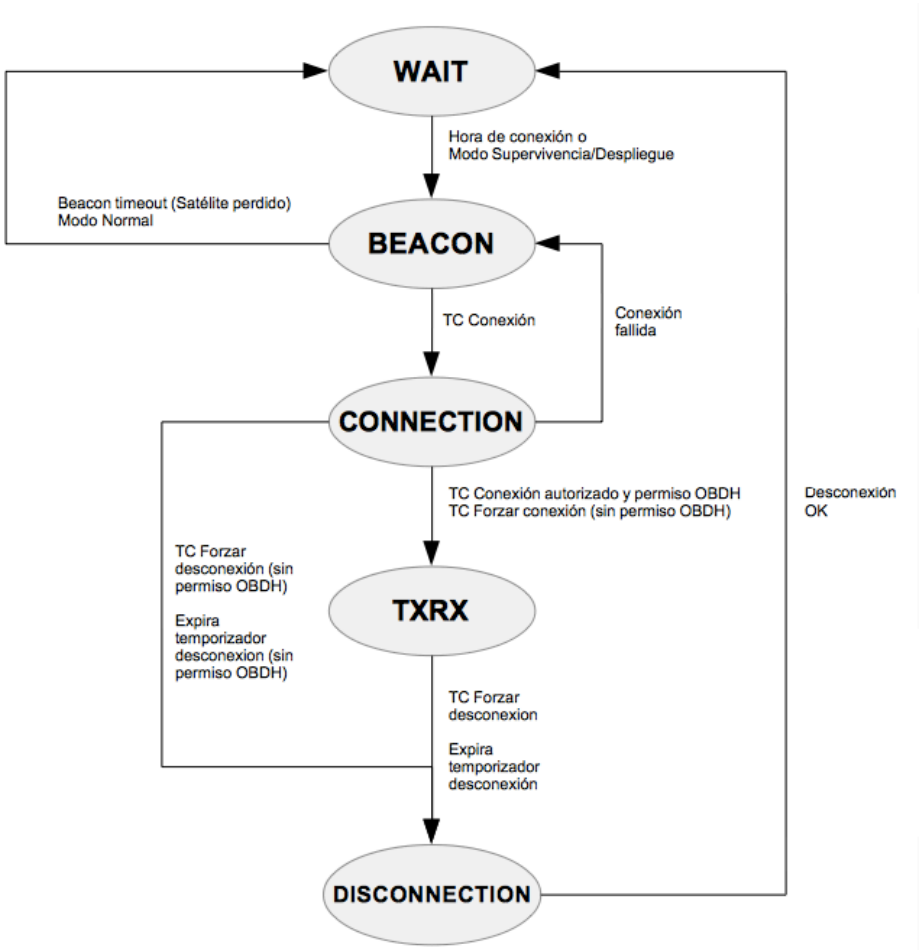

Figura 5.12 Diagrama funcional del sistema de comunicaciones [3].

ejecutar a los demás y por lo tanto acapara todo el tiempo de procesamiento. Esto se consigue mediante un hilo con una baja prioridad cuyo única función es refrescar el contador del WDT.

El "hard WDT" viene implementado en el Nanopower. Consiste en un corte del suministro eléctrico si se produce una caída de la tensión por culpa de la descargas de las baterías. Este hard reset afecta a todas las conexiones, ya estén bajo el efecto del WDT o estén conectadas a los puertos de carga fija.

5.8.1 Diagrama funcional del reporte de errores

Tal y como puede observarse en el diagrama de la Figura 5.18, el sistema de gestión de los errores recoge los returns de las funciones más críticas de la plataforma, generando una estructura con un identificador y enviándola a la cola de mensajes hacia el segmento terreno [4].

5.8.2 Diagrama funcional del chequeo de subsistemas

Cuando se produce un reinicio del sistema, tal y como se ejemplifica en la Figura 5.19, se realiza un chequeo de los subsistemas. Al inicializarse el OBDH se crea un hilo que reclamará el estado de los subsistemas uno a uno, guardando en una estructura los resultados. Cuando finaliza, se revisa la estructura en busca de algún código de error; si se encuentra, se envía al sistema de comunicación junto con un identificador para dar una prioridad elevada. Si no se observa ningún funcionamiento anómalo se desecha el reporte y se elimina el hilo.

5.9 Gestión de los telecomandos

La gestión de comandos es una parte vital para el desarrollo de la misión [17]. A continuación se nombran los telecomandos establecidos para la plataforma según su pertenencia al sistema de comunicación , cargas

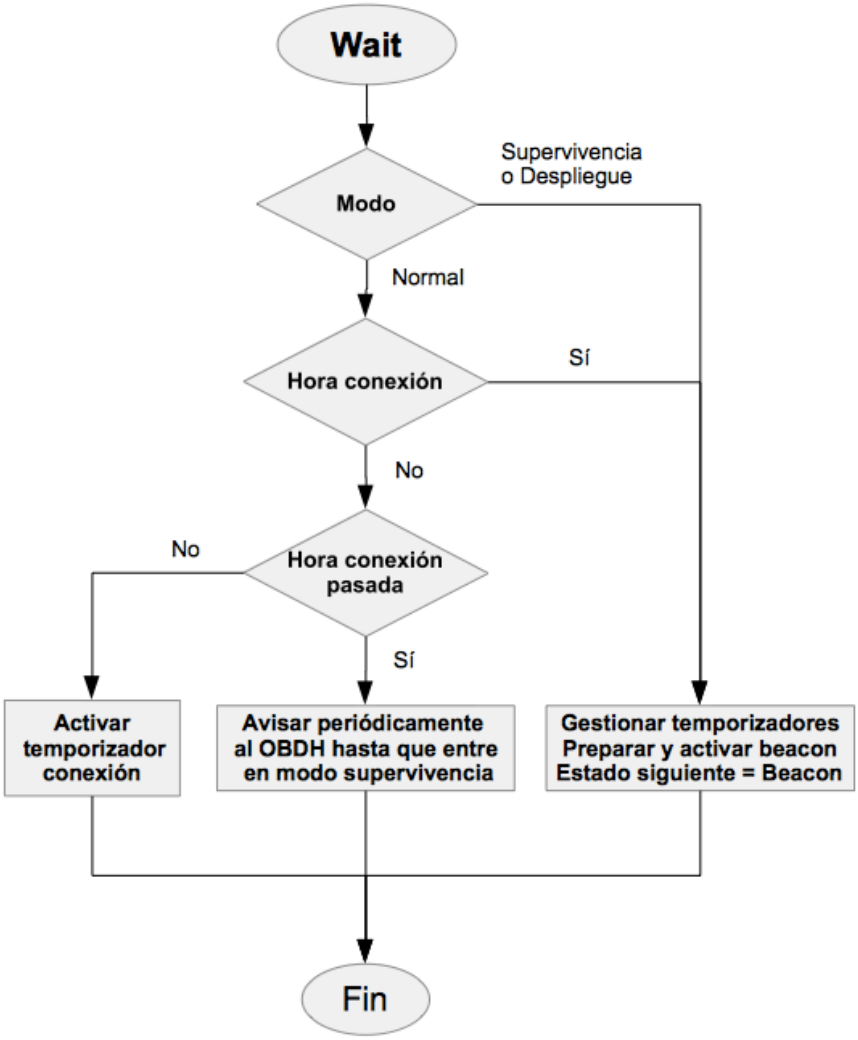

Figura 5.13 Diagrama funcional del estado de espera [3].

de pago u OBDH.

5.9.1 Telecomandos generales para el sistema OBDH

- Puesta en hora del reloj en tiempo real interno.
- Reboot del satélite.
- Petición de estado del subsistema EPS.
- Petición de estado del subsistema experimentos.
- Petición del log del sistema.
- Modificación del estado del satélite, pudiendo forzar un modo durante un periodo de tiempo.
- Petición de estado general del satélite.

5.9.2 Telecomandos generales para el sistema de comunicación

- Apagado de las comunicaciones.
- Petición de estado del subsistema de comunicaciones.
- Envío de datos.

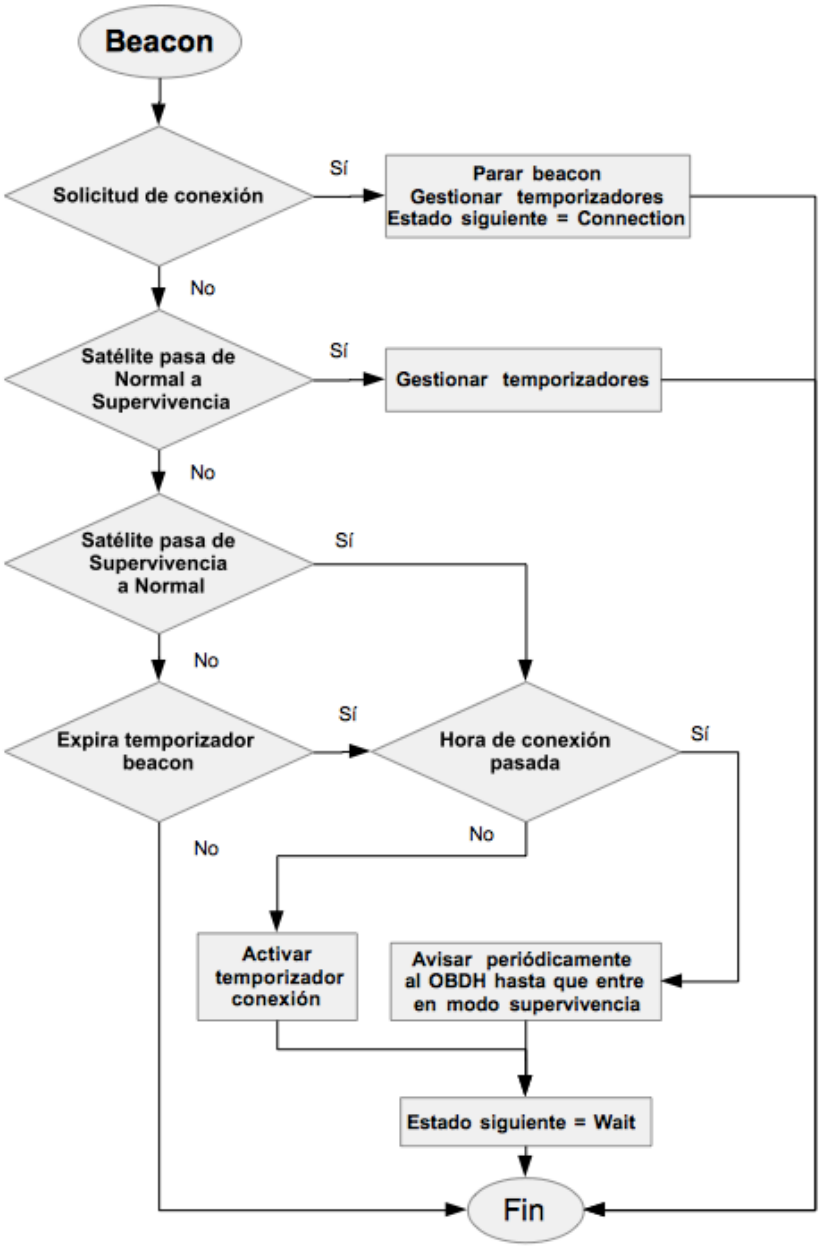

Figura 5.14 Diagrama funcional del modo baliza [3].

- Cancelar comunicación.
- Cambiar parámetros de la comunicación.

5.9.3 Telecomandos generales para las cargas de pago

- Generación de experimento programado.
- Anulación de experimento programado.

5.10 Gestión de memoria no volátil

El empleo de memorias no volátiles en entornos espaciales tiene una serie de inconvenientes debido a la alta posibilidad de fallos como consecuencia de la radiación incidente. Debido a este fenómeno, puede producirse

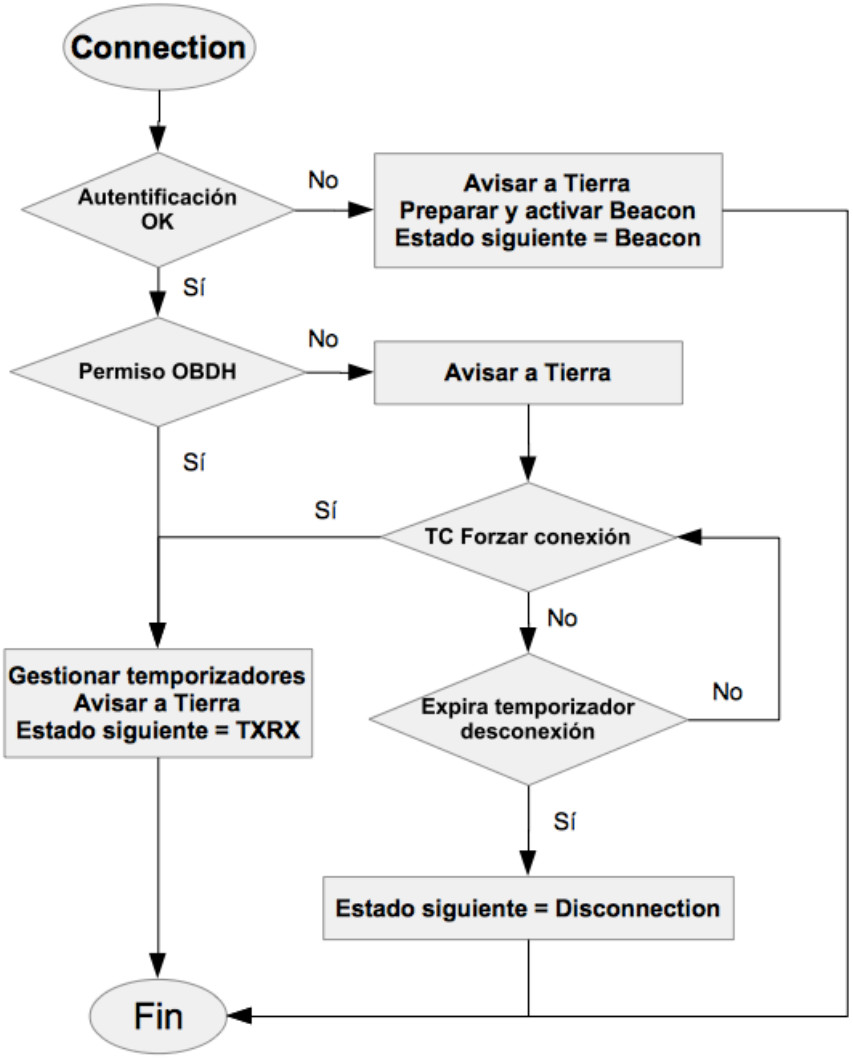

Figura 5.15 Diagrama funcional del modo conexión [3].

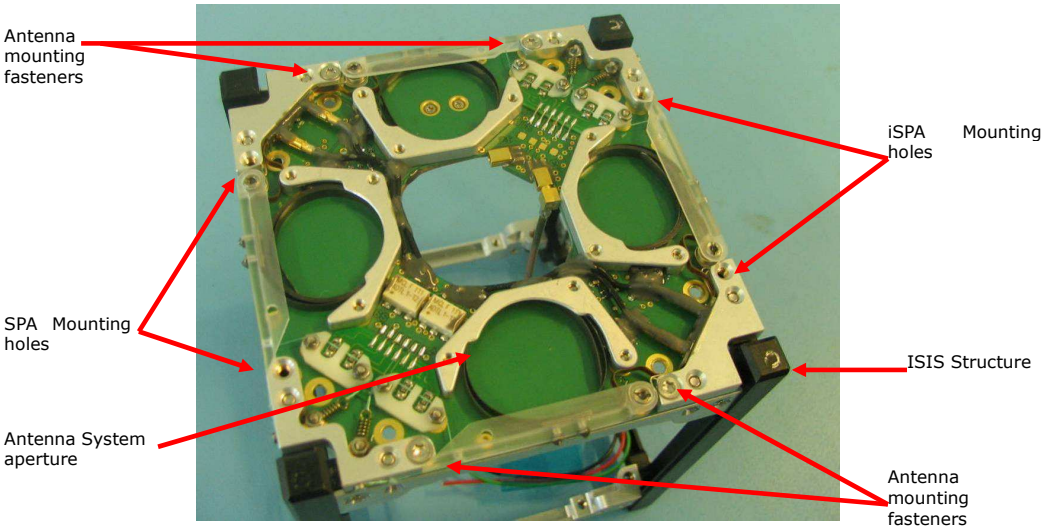

Figura 5.16 Dispositivo de despliegue de antenas.

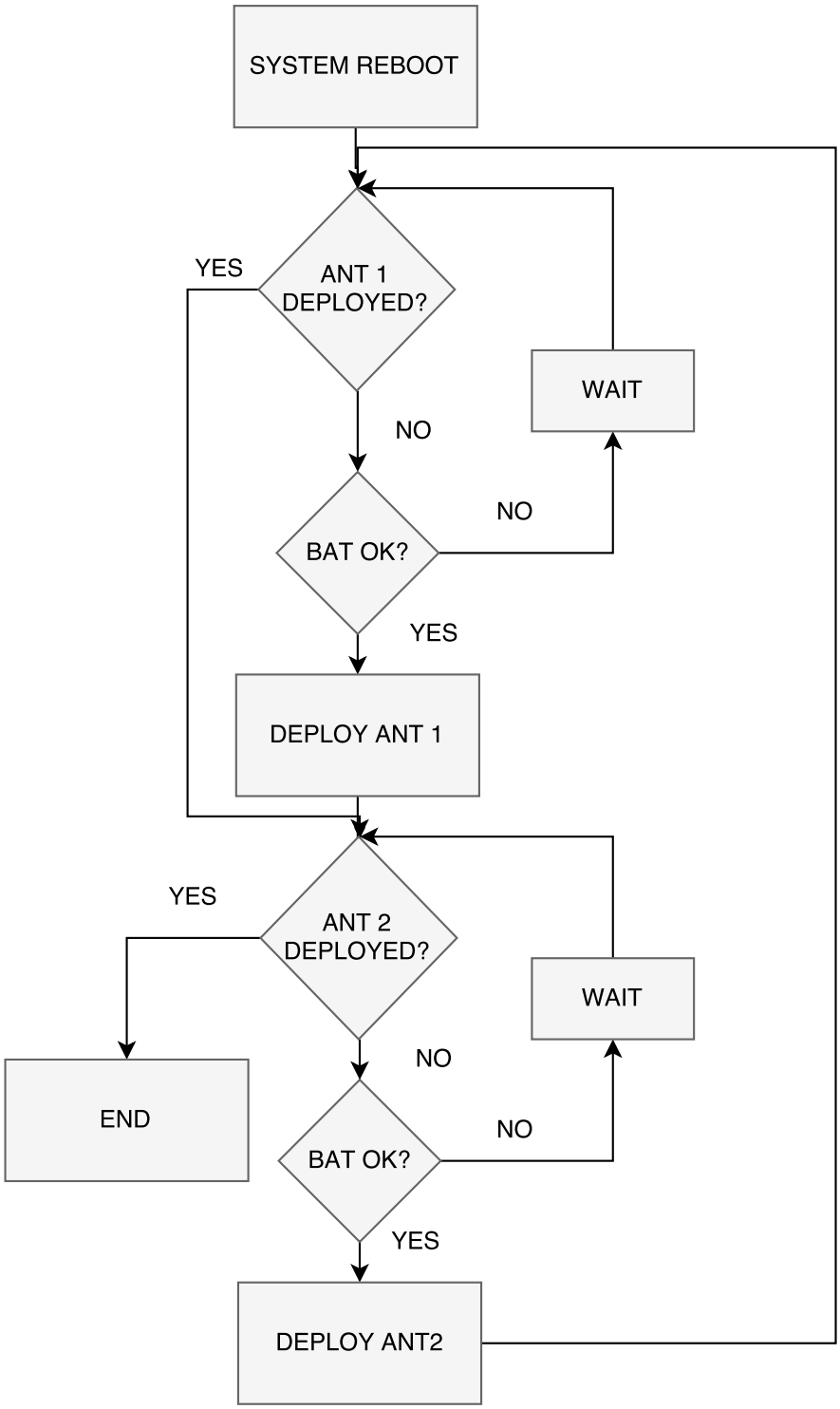

Figura 5.17 Diagrama funcional del modo conexión.

por ejemplo un cambio en el estado de un bit y corromperse el archivo guardado. Por esto es necesario un algoritmo de control de fallos que detecte y evite zonas defectuosas de la memoria.

5.10.1 Memoria no volátil de la placa Nanomind

La placa Nanomind A712C presenta una memoria flash no volátil de 4 MB de almacenamiento para datos. El dispositivo es el modelo AT49BV320DT de la marca Atmel, con dos megas de posiciones de 16 bits. Presenta 63 bloques de 64K bytes y 8 de 8K bytes. El tiempo de escritura es de 10 μs, el de acceso 10 ns y el

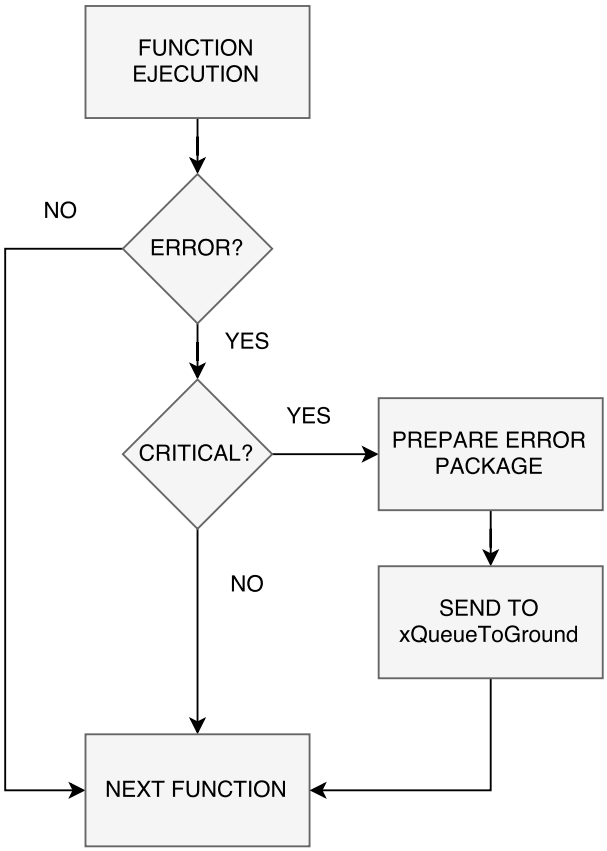

Figura 5.18 Diagrama funcional del reporte de errores.

borrado 100 ms.

En las librería proporcionadas por Gomspace no se encuentra la librería *libstorage*, en la que se define un sistema de ficheros para la gestión de datos en la memoria. Ha sido por lo tanto necesario el desarrollo de una interfaz que gestione la escritura y borrado de los datos.

Esta interfaz guarda las variables que se le proporcionan junto con su longitud y checksum. Este último dato servirá para la verificación de que los archivos guardados no han sido corrompidos. Cada vez que se lea un archivo, este se extraerá de la memoria junto con su checksum, se volverá a calcular un checksum de la extracción y se comparará que coincide con el que se tiene guardado.

En cuanto a la utilización de la memoria se ha optado por emplear para cada experimento un sector de la misma. No es la forma más eficiente de gestionar la memoria pero sí la más robusta. Además para cada experimento se reservan hasta 64 Kbytes de información, lo que es más que suficiente dado la naturaleza de los mismos.

5.10.2 Diagrama funcional del chequeo de memoria

Para asegurar el correcto funcionamiento de la memoria y la integridad de los archivos, cada vez que se produce un reinicio del sistema, el OBDH realiza un test a la memoria no volátil. Este consiste en la lectura de los archivos de cada bloque, el cálculo de su checksum y su posterior comprobación con el que se guardó. Si en algún bloque se produce una no coincidencia, este será borrado para evitar el uso de estos datos, Figura 5.20.

5.11 Gestión del sistema ADCS

El sistema ADCS es el enargado de controlar la actitud del satélite y realizar el detumbling una vez que se ha separado del P-POD [18]. Está formado por tres magneto-pares ensamblados por la empresa ISIS y gestionados mediante la plataforma OBDH. Estos tres componentes consiguen que el cubesat tenga control en sus tres ejes y por lo tanto pueda modificar su actitud en el ángulo deseado [1].

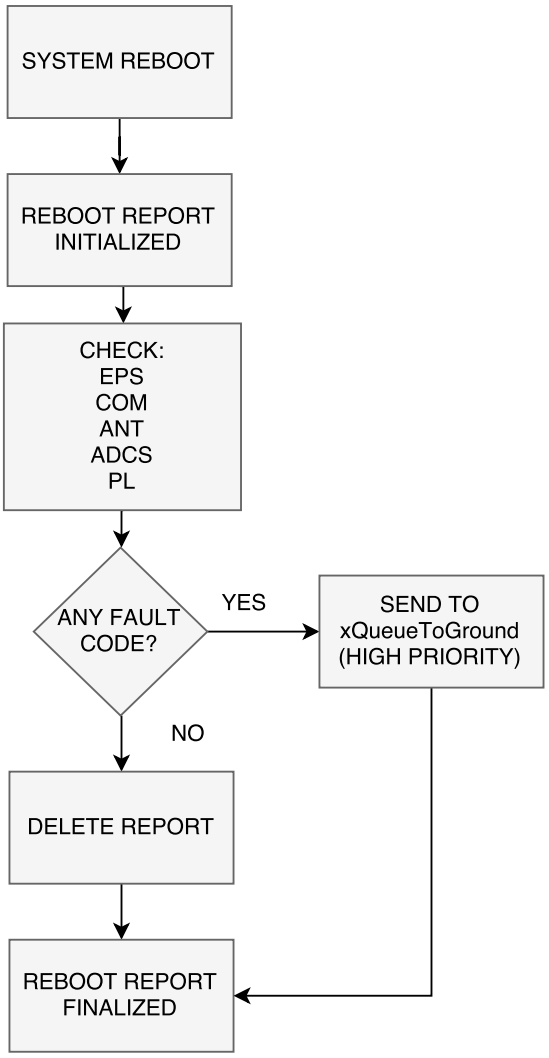

Figura 5.19 Diagrama funcional del chequeo de subsistemas.

Como el mecanismo de funcionamiento de los magneto-pares o magneto-torques está basado en la fuerza que produce un campo magnético sobre una corriente como la que circula en estos dispositivos, la orientación del campo magnético terrestre tendrá un papel fundamental en la resultante de momentos en la plataforma. Para conseguir esta información, la plataforma Nanomind viene equipada con magnetómetros, giróscopos y acelerómetros, sensores que ayudarán a conocer la orientación respecto del campo magnético y el cambio de actitud. Posee 3 rods (dipolos), uno por eje (X, Y, Z), y son actuados mediante señales PWM a 5V [12].

Al actuar sobre el magneto-torque genera un vector de momento magnético (m) en el cuerpo del satélite. Considerando el campo magnético de la tierra (B), que se calcula con el magnetómetro, es posible calcular el par de fuerzas (T) necesario para llevar a cabo el control de actitud del satélite .

$$\vec{T} = \vec{m} \wedge \vec{B} \tag{5.1}$$

Respecto a los sensores magnéticos son de 3 ejes que se gestionan internamente al sistema, mediante el bus I2C. El sensor tiene como salidas las siguientes variables:

- Componente X del campo magnético, en mG.
- Componente Y del campo magnético, en mG.
- Componente Z del campo magnético, en mG.

En el proceso de detumbling se buscará reducir el spinning del satélite en los tres ejes y por debajo de un nivel estipulado. Podría por tanto considerarse que en esta tarea, el control consistirá en 3 controles independientes, uno por eje, en el cual cada uno buscará detener el spin hasta un mínimo.

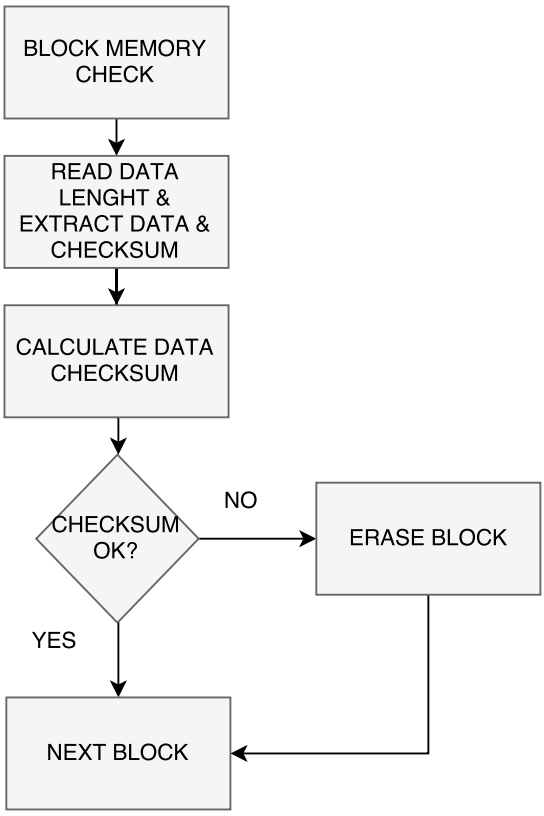

Figura 5.20 Diagrama funcional del chequeo de memoria.

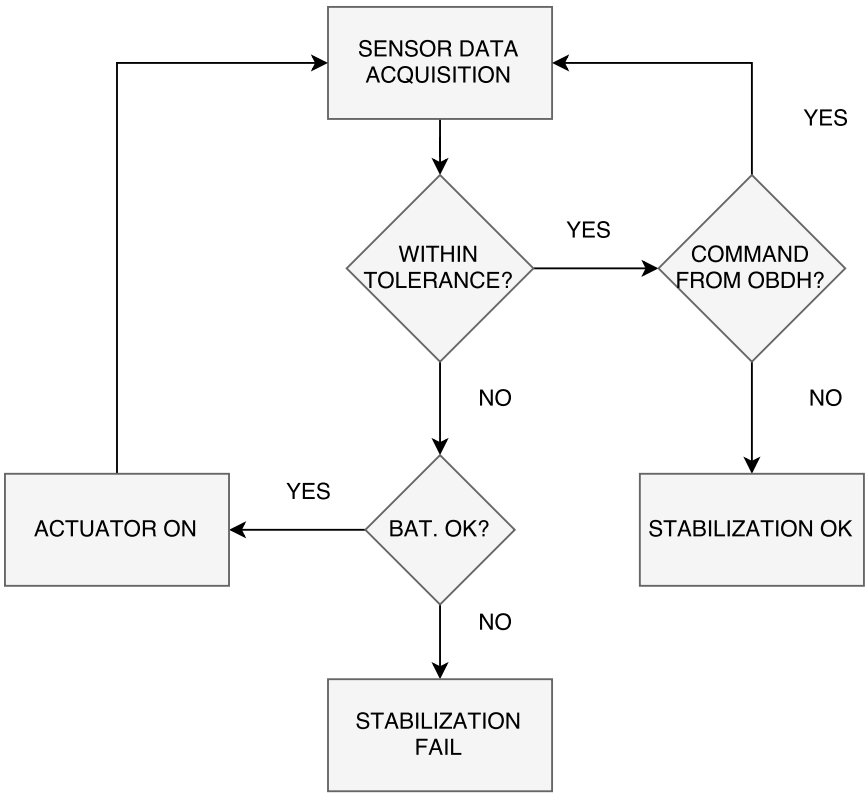

Figura 5.21 Diagrama funcional del sistema ADCS.

Como se puede observar en la Figura 5.21, tras una lectura de los sensores, el OBDH comprueba que los parámetros de los giróscopos y acelerómetros están dentro de tolerancia. Si esto es así, comprueba que no exista un telecomando específico para modificar la actitud del satélite y da por buena la estabilización. Hay que tener en cuenta que los telecomandos desde el segmento terreno son tomados como una orden absoluta, obviando los niveles de batería de la plataforma.

En el caso de que no esté dentro de tolerancia los actuadores seguirán la ley de control establecida para evitar el giro descontrolado del satélite, siempre y cuando se disponga suficiente nivel de energía para realizarlo. Cuando se llega a un una estabilización fallida o satisfactoria, se vuelve a ejecutar el hilo desde la lectura de los sensores [14].

5.12 Gestión de los experimentos

Este bloque es el que presenta una prioridad menor, ya que solo se encontrará activo si el satélite se encuentra en condiciones seguras de operación y con suficiente carga en las baterías. Se gestionarán los permisos del bloque de experimentos de acuerdo con los niveles de energía disponibles.

Para este proyecto se han realizado las interfaces necesarias para las cargas de pago, pero se han considerado en todo momento como unas cargas de pago genéricas, de forma que el diseño del software de la plataforma pueda ser trasladado a otras misiones con tamaño y demanda de potencia y datos semejante.

Para la gestión de la planificación de los experimentos es fundamental el empleo de un reloj en tiempo real como el que incorpora la placa Nanomind.

5.12.1 Reloj en tiempo real de la placa Nanomind

La placa Nanomind dispone de un reloj en tiempo real. Es un pequeño chip de bajo consumo con un oscilador de 32,768 kHz el cual opera mediante un sistema de carga con un diodo que alimenta a un condensador de tantalio con suficiente capacidad para mantener el reloj funcionando hasta una hora después del apagado de la placa.

Durante el encendido de la placa la hora es leída del RTC y la función *clock_set_time* es llamada para ajustar el reloj del sistema. Una vez hecho esto, la hora es actualizada por medio de un contador de ticks del sistema operativo. Si hay una discrepancia entre los relojes o un fallo en el reloj del sistema la única forma de tomar de nuevo la hora del RTC es reiniciando el sistema.

Esta funcionalidad de la placa es crítica a la hora de planificar los experimentos a realizar por la plataforma y conocer el tiempo restante hasta volver a tener conexión con el segmento terreno.

6 Funciones e interfaces

En este capítulo se presentan las funciones principales implementadas en el software del satélite y las interfaces entre el sistema de comunicación y el OBDH. Estas funciones han sido clasificadas según el tipo de gestión que realicen siguiendo la distribución del capítulo anterior.

6.1 Configuración de hilos

En la Figura 6.1 se presentan los diferentes hilos y sus relaciones mediante colas. En el software existen cinco colas: dos entre el hilo de telecomandos y el subsistema de comunicaciones, dos internas del sistema de comunicaciones y una para verter los datos al hilo de gestión de errores. Las colas destinadas al sistema de cargas de pago no han sido implementados para este proyecto, ya que no se ha desarrollado aún la interfaz entre ambas. No debe confundirse la ausencia de colas con la ausencia de interacción entre sistemas; por ejemplo, el hilo de gestión de la potencia puede parecer aislado, pero esto no significa que los demás hilos no pueden utilizar funcionalidades del EPS

El hilo de gestión de telecomandos sirve de enlace entre el subsistema de comunicaciones y el de cargas de pago, sirviéndose de funciones de otros subsistemas para implementar los distintos telecomandos de los que dispone la plataforma. Los errores son reportados por todos los hilos al de gestión de errores, que se encargará de generar informes para descargar al segmento terreno. Dentro del sistema de comunicación, el hilo COMData prepara datos y se los entrega al hilo COM, que controla la comunicación con tierra.

En el propio código, dentro de los ficheros de cada hilo, se disponen de funcionalidades específicas de los sistemas hardware a los que se encuentran asociados. En el caso del hilo del sistema ADCS, en el archivo fuente se tienen funciones para realizar el HK cuando el hilo de telecomandos lo solicita.

El hilo deployment, encargado de la gestión del despliegue de las antenas y su HK, se elimina en cada encendido cuando detecta que se ha producido de forma satisfactoria el despliegue. Cuando se lleve a cabo un reinicio, se creará y comprobará de nuevo el estado.

En la Figura 6.2 se ha presentado un diagrama funcional que refleja de forma general el inicio de las tareas en el encendido de la plataforma. Se crea la cola para los errores, ya que se realizará un test al reinicio de los subsistemas y es necesario volcar estos datos en una cola hasta que el sistema de comunicaciones esté operativo para enviarlo. El hilo de gestión de potencia es el primero en crearse, fijando el estado del satélite en modo seguro. A continuación se crean las colas con el sistema de comunicaciones y los hilos restantes del sistema.

Cabe destacar, que antes de que se inicie este proceso se lleva a cabo una cuenta atrás si se reconoce el encendido como el primero que realiza el satélite. Esto es necesario para evitar situaciones indeseadas en la fase de comissioning, una vez que se ha liberado del lanzador. Este tiempo puede ser fijado por el usuario, siendo de 45 minutos por defecto.

6.2 Gestión de la energía

Para la gestión de la energía se tiene un archivo de cabecera en el que se definen las funciones principales que gobiernan el sistema. En el primer caso se definen los niveles de tensión y temperatura a partir de los cuales se realiza el cambio de un estado del satélite a otro. Estos límites dispondrán de un margen de histéresis para evitar que el satélite cambie de estado de forma oscilatoria y descontrolada. A continuación se define

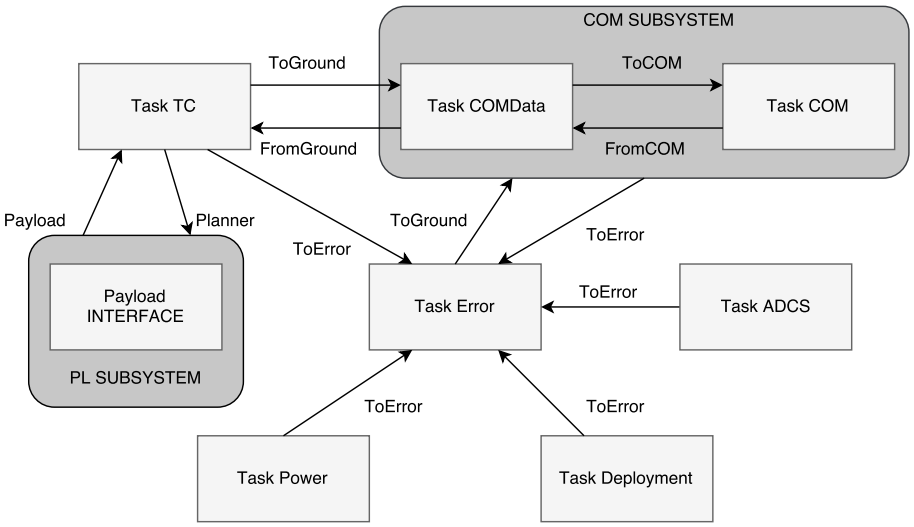

Figura 6.1 Esquema de la arquitectura de hilos.

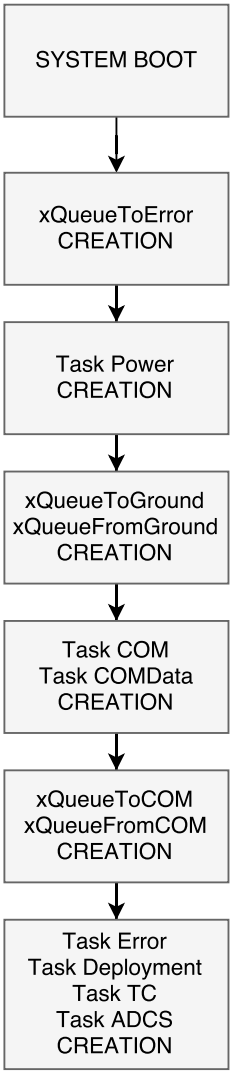

Figura 6.2 Proceso de inicio de la plataforma.

una variable tipo *enumerate* que guardará el estado en el que se encuentra la plataforma y una etiqueta para referenciar al hilo de gestión de energía.

```
/*
* Definitions for EPS configuration
*/
#define EPS_INDEX 0     //CSP index
#define MIN_V_EXP 14600   // minimum voltage level for exp
#define MIN_V_NOMINAL 15000  // minimum voltage level for nominal mode
#define MIN_T_NOMINAL -20   // minimum voltage level for nominal mode
#define MAX_T_NOMINAL 60   // minimum voltage level for nominal mode

/*
* Satellite state type declaration
*/
typedef enum {
  NOMINAL,   //Nominal Mode
  SAFE     //Safe Mode
}state;

state satellite_state; // satellite state variable

/*
* Setting Handle for vTaskPower to suspend the task
*/
xTaskHandle xTaskPowerHandle;
```

En el mismo archivo se tiene la definición del módulo de gestión de potencia. Esta función no solo ejecuta el hilo del sistema de potencia, si no que inicializa las conexiones I2C entre el hardware del Nanomind y el Nanopower además de las diferentes variables y control de errores.

```
/*
 * @func: start eps subsistem, including I2C bus, vTaskPower threat and power
     supervisor
 * @return:--
 */
void task_power_init(void);
```

El hilo en sí viene definido de la siguiente forma:

```
/*
 * @func: task in charge of power supervisor, rtc checking, temperature and
     current
 * @in: pointer to a parameter (not used). Refer to freeRTOS Manual
 * @return:-
 */
void vTaskPower(void * param);
```

6.3 Gestión de la comunicación

Para el sistema de comunicaciones se presenta la interfaz con el sistema OBDH de la plataforma, así como el hilo de despliegue de las antenas. El interfaz y el módulo de comunicaciones que aparece en el código de la plataforma ha sido realizado en el Trabajo Fin de Grado [3], reflejado en la bibliografía y disponible en la base de datos de la Universidad de Sevilla.

6.3.1 Interfaz con el sistema de comunicaciones

En el archivo de cabecera se disponen las definiciones de las variables y de las funciones. Se define una variable que será el identificador de los mensajes que se enviarán y recibirán por medio de las colas entre el sistema de comunicaciones y el OBDH.

```
/* Message type exchanged between OBDH and COM*/

typedef enum message_type_t {

  /* FROM OBDH TO COM */
  SEND_EXP = 0,      //Exp data available
  SEND_HK = 1,       //HK from COM request
  SEND_TC_RESPONSE = 2,  //response to TC sent from ground

  /* FROM COM TO OBDH */
  TC = 3,          // TC to execute
  ID_STATUS_STORED = 4,  //ID stored y pending TX
  ID_STATUS_DISCARDED = 5, //ID discarded before send
  ID_STATUS_SENT = 6,   //ID sent and received
  SAT_LOST = 7,      //Unable to connect

} message_type_t;
```

El identificador anterior formará pate de una estructura mayor, en la que también se tendrá la longitud, el puntero a los datos que quieren transferirse y una variable bandera para liberar memoria en el sistema.

```
/* Message exchanged between OBDH and COM */
typedef struct message_t {

  message_type_t type;    // Message type
  unsigned short id;    // Number of message
  unsigned short data_length;  //data bytes
  unsigned char * data;   // pointer to data
  unsigned short * free_flag;  // flag for free memory

} message_t;
```

Una de las variables fundamentales en la comunicación entre ambos sistemas es el permiso del OBDH para establecer comunicación con el segmento terreno.

```
/* Permission to communication with the ground segment
 * @return: permision code
*/
int com_permission;
```

Las siguientes líneas de código establecen los identificadores para las dos colas de comunicación. La primera para los datos que quieren enviarse al sistema de comunicación, ya sea para información interna o para enviar datos a tierra, y otra en el sentido inverso.

```
/* Communication queues between COM and OBDH */
xQueueHandle xQueuetoCOM;  // queue from OBDH to COM
xQueueHandle xQueuefromCOM;  //queue from COM to OBDH
```

6.3.2 Despliegue de las antenas

Para el sistema de despliegue de las antenas se define una estructura de configuración del satélite para futuros desarrollos en los que se adapte este proceso teniendo en cuenta el estado de la plataforma.

```
/*
*Cepheus configuration structure
*/
typedef union __attribute__ ((__packed__)) _cepheus_config
{
  unsigned char raw[13]; /**< Unformatted Time TC */
  struct __attribute__ ((__packed__))
  {
    unsigned char config_id; //ID for config number
    unsigned int data_length; //in bytes
    int data;        // data attached to configuration state
    unsigned int checksum;
  } fields; /**< Struct with individual fields*/
} cepheus_config;
```

Con la siguiente función se inicializan las comunicaciones con el sistema por medio del bus I2C, además de configurar el reporte de errores y activar el hilo propio de la tarea.

```
/*
* @func: initialize ant deployment process, including task for deployment
*/
void deployment_init(void);
```

Por último, el hilo específico para el despliegue de las antenas. Este hilo desaparecerá en cuanto se comprueben que el proceso se ha realizado correctamente. Esto ayuda a no sobrecargar el sistema con tareas que no son imprescindibles y por lo tanto estarían ocupando tiempo de procesamiento y memoria.

```
/*
* @func: initialize deployment task
* @in: pointer to a parameter (not used). Refer to freeRTOS Manual
*/
void vTaskDeployment(void* parameters);
```

6.4 Gestión de errores

Se presentan a continuación la definición de errores empleadas en el software del satélite:

```
/*
 * Error definitions
 */
#define NO_ERROR        0x01 //
#define LOW_EPS_VOLTAGE      0x10 // Battery voltage below set value
#define HIGH_EPS_VOLTAGE     0x11 // Battery voltage above set value
#define HIGH_EPS_TEMPERATURE  0x12 // Battery temp above set value
#define LOW_EPS_TEMPERATURE  0x13 // Battery temp below set value
#define UNSET_RTC_TIME     0x14 // RTC time lost
#define RTC_FAILURE       0x15 // RTC failure in taskpower supervision

#define ERROR_INIT        0x50
#define ERROR_GET_SEMAPHORE_FAILED 0x51
#define ERRORE_INDEX_ERROR    0x52
#define ERROR_INDEX_ERROR   0x53 /**< Incorrect index specified */
#define ERROR_BITRATE_INPUT_ERROR  0x54 /**< Failed to set the bitrate of the
    TRXUV */
```

```
#define ERROR_CWCHAR_INPUT_ERROR 0x55 /**< Failed to set the cwcharrate of the
    TRXUV */
#define ERROR_IDLE_STATE_ERROR    0x56 /**< Failed to go into idle mode of the
     TRXUV */
#define ERROR_OUTPUT_MODE_ERROR 0x57 /**< Failed to go into outmode of the
    TRXUV */
#define ERROR_TRXUV_COMPONENT_ERROR 0x58 /**< Failed to choose a device in the
    TRXUV */
#define ERROR_WRONG_ADC_CHANNEL 0x59 /**< Failed to choose a correct ADC
    channel TRXUV */
#define ERROR_RESET_SYSTEM    0x5A /**< Failed to reset both microcontrollers
    in the TRXUV */
#define ERROR_MEM_ALLOC    0x5B /**< Failed to allocate memory in the TRXUV */
#define ERROR_ATT_ERROR    0x5C /**< Failed to set attenuation value in the TXS
     */
#define ERROR_PARAM_OUTOFBOUNDS 0x5D /**< Failed to set the correct number of
    parameters */
#define ERROR_OUTPUT_OUTOFBOUNDS 0x5F /**< Failed to set a correct output
    variable */
#define ERROR_NOT_DEFINED    0xFF

/*
 * Subsystem error id definition
 * Initialization error in i2c bus
 */

#define I2C_BUS_INITIALIZATION 0x60
#define EPS_INITIALIZATION  0x61
#define TRXUV_INITIALIZATION 0x62
#define ANT_INITIALIZATION  0x63
#define MTQ_INITIALIZATION  0x64
#define SAFE_MODE_CAUSE  0x65
```

Una de las comprobaciones realizadas dentro de la gestión de los errores es el estado del reloj en tiempo real. Cada cierto tiempo o cada vez que se reinicie el sistema se ejecutará la función inferior, cambiando el estado del satélite según el return e informando al sistema de comunicaciones que el tiempo de referencia es correcto o no.

```
/*
* @func: compare RTC and reference time to know if a RTC problem exists
 * @return: 1 if OK, -1 if RTC is not working (its value is 0), 0 if RTC has
     been reset
 */
int test_time_boot();
```

Se genera una cola para que todos los errores críticos que se produzcan en el funcionamiento de la plataforma sean gestionados por el hilo de errores. Cuando se detecta un error, este se deposita en la cola con la información adjunta para que el sistema determine si se envía a tierra o se elimina la información.

```
/*
* Queue used to send error to COM (Use only major errror)
*/
xQueueHandle xQueuetoERROR;
```

Para la realización de los informes de estado de la plataforma al reinicio de la misma, es necesario establecer las colas necesarias para la gestión de la información antes de inicializar las conexiones con el bus I2C.

```
/*
*@func: initialize queues to provide boot system report
* MUST be set at the beginning of boot process
*/
void error_management_init(void);
```

La creación del informe de errores en el reinicio se realiza con la función inferior, a la que hay que introducirle un puntero a una estructura tipo para la gestión de errores y otra para el mensaje que creará y enviará al sistema de comunicación.

```
/*
*@func: generate boot report of every subsystem
* @in1: pointer to error structure
* @in2: pointer to error message structure
* @return: fault code
*/
unsigned short generate_boot_report(error_data* error_message,boot_error_report
    * boot_report);
```

Aparte del informe inicial, se inicializa la gestión de errores para el funcionamiento nominal de la plataforma.

```
/*
* @func: initialize error management: classification of error, prepare packets
    to sent to COM,
* analyze boot error report, create task error.
*/
void task_error_init(void);
```

El hilo específico en freeRTOS ha de definirse como:

```
/*
* @func: task error declaration
 * @in: pointer to a parameter (not used). Refer to freeRTOS Manual
*/
void vTaskError(void* param);
```

Para la generación del informe de errores de los subsistemas al reinicio del OBDH se tiene la siguiente función, donde la entrada es un puntero a la dirección de memoria donde se desea guardar la información.

```
/*
* @func: generate error report; test subsystems connection during boot process
* @in: pointer to data struct error_message type
*/
void generate_error_report(error_data* error_message);
```

6.5 Gestión de los telecomandos

En la gestión de los telecomandos se definen las identificaciones para las diferentes opciones:

```
/*
* Definition of TC types
*/
#define RTC_SET       0x20 // Set real time of the satellite
#define CPU_RESET     0x10 // Reset Nanomind (soft)
#define HK_EPS_REQUEST   0x30 // EPS HK to download to ground
```

```
#define FORCE_STATE     0x40 // Force satellite state
#define SATELLITE_STATE_AUTO  0x50 // Recover automated satellite state
#define EPS_SOFT_RESET   0x60 // Force EPS soft reset
#define EPS_HARD_RESET   0x70 // Force EPS hard reset
#define EPS_CONFIG_SET   0x80 // Configure EPS
#define EPS_CONFIG_GET   0x90 // Get EPS configuration
```

A continuación se presentan las distintas variables definidas para gestionar los distintos tipos de telecomandos. En todas se presenta el tiempo en el que se recibió el telecomando, o en el que se produjo el error, seguido de un identificador para conocer la respuesta del satélite.

```
/*
* Declaration of time structure
*/
typedef union __attribute__ ((__packed__)) _tc_time
{
  unsigned char raw[8]; /**< Unformatted Time TC */
  struct __attribute__ ((__packed__))
  {
    uint32_t time_sec;
    uint32_t time_nsec;
  } fields; /**< Struct with individual fields*/
} tc_time;

/*
* Declaration of response to TC force state command
*/
typedef union __attribute__ ((__packed__)) _tc_force_response
{
  unsigned char raw[6];
  struct __attribute__ ((__packed__))
  {
    unsigned char tc_error; // Error ID
    unsigned char satellite_st; //New state
    uint32_t time_sec;  //Time of execution
  } fields;
} tc_force_response;

/*
*Declaration of standard TC response
*/
typedef union __attribute__ ((__packed__)) _tc_error
{
  unsigned char raw[9];
  struct __attribute__ ((__packed__))
  {
    unsigned char errortc; // Error ID
    uint32_t time_sec;  //Time of TC execution
    uint32_t time_nsec;
  } fields;
} tc_error;
```

En cuanto a la estructura para gestión de la hora, cabe destacar que el tiempo absoluto en el sistema es representado empleando el formato de tiempo POSIX, un sistema para la descripción de instantes de tiempo definido como la cantidad de segundos transcurridos desde la medianoche UTC del 1 de enero de 1970, sin contar segundos intercalares. En la librería C se emplea el siguiente tipo de dato para la hora:

```
/*
* Time structure declaration
*/
typedef struct __attribute__((packed)){
  uint32_t tv_sec;
  uint32_t tv_nsec;
}timestamp_t;
```

Por lo tanto en la estructura aparece la hora en segundos desde el uno de enero de 1970, además de nanosegundos.

Cuando se produce un reinicio o encendido del sistema, si no se tiene habilitado el RTC, el reloj del sistema se inicializa en cero y comienza a actualizarse con los ticks del sistema operativo. Esto quiere decir que la fecha de inicio es el 1 de enero de 1970.

El hilo específico para la gestión de los telecomandos viene definido por:

```
/*
 * @func: TC task definition. In charge of TC management and response to COM
 * @in: pointer to a parameter (not used). Refer to freeRTOS Manual
 */
void vTaskTC(void* parameters);
```

Para inicializar las colas y las variables del módulo de gestión de los telecomandos es necesario ejecutar la función:

```
/*
 * @func: TC subsystem initialization. Creation of queues between COM and OBDH
 */
void tc_init(void);
```

Se ha implementado una función para modificar el tiempo del sistema desde el segmento terreno. Como entrada se introduce el tiempo a modificar, la longitud de los datos en el telecomando y la identificación del mismo.

```
/*
 * @func: function to modified system time by TC
 * @in1: new system time
 * @in2: length of TC data
 * @in3: TC ID
 */
void set_time_tc(unsigned char* timetc,unsigned short tc_length,unsigned short
    tc_id );
```

La siguiente función resetea mediante software el subsistema OBDH de la plataforma, devolviendo el identificador de error y el tiempo del reinicio:

```
/*
 * @func: Function for OBDH reboot
 * @in1: TC ID
 */
void obdh_reboot(unsigned short tc_id);
```

Para el envío del HK del sistema EPS se dispone de la función:

```
/*
* @func: function for HK request from ground
*/
void hk_eps_request(unsigned short tc_id);
```

La estructura de HK que devuelve este telecomando se puede encontrar también en el mensaje enviado en el beacon, el cual contiene las siguientes magnitudes:

```
/*
 * Union for storing the block of telemetry values coming from the EPS. HK
     version 2.
 */
typedef union __attribute__ ((__packed__)) _gom_eps_hk_t
{
  unsigned char raw[133]; /**< Unformatted GOM EPS telemetry */
  struct __attribute__((packed))
  {
    unsigned short commandReply; /**< reply of the last command */
    unsigned short vboost[3]; /**< Voltage of boost converters [mV] [PV1, PV2,
        PV3] */
    unsigned short vbatt; /**< Voltage of battery [mV] */
    unsigned short curin[3]; /**< Current in [mA] */
    unsigned short cursun; /**< Current from boost converters */
    unsigned short cursys; /**< Current out of battery */
    unsigned short reserved1; /**< Reserved for future use */
    unsigned short curout[6]; /**< Current out [mA] */
    unsigned char output[8]; /**< Status of outputs */
    unsigned short output_on_delta[8]; /**< Time till power on */
    unsigned short output_off_delta[8]; /**< Time till power off */
    unsigned short latchup[6]; /**< Number of latch-ups */
    unsigned int wdt_i2c_time_left; /**< Time left on I2C wdt */
    unsigned int wdt_gnd_time_left; /**< Time left on I2C wdt */
    unsigned char wdt_csp_pings_left[2]; /**< Pings left on CSP wdt */
    unsigned int counter_wdt_i2c; /**< Number of WDT I2C reboots */
    unsigned int counter_wdt_gnd; /**< Number of WDT GND reboots */
    unsigned int counter_wdt_csp[2]; /**< Number of WDT CSP reboots */
    unsigned int counter_boot; /**< Number of EPS reboots */
    short temp[6]; /**< Temperature sensors [0 = TEMP1, TEMP2, TEMP3, TEMP4,
        BATT0, BATT1] */
    unsigned char bootcause; /**< Cause of last EPS reset */
    unsigned char battmode; /**< Mode for battery [0 = normal, 1 = undervoltage
        , 2 = overvoltage] */
    unsigned char pptmode; /**< Mode of PPT tracker */
    unsigned short reserved2;
  } fields; /**< Struct with individual fields of GOM EPS telemetry. HK version
       2. 92 */
} gom_eps_hk_t;
```

Desde tierra puede forzarse al satélite a un estado aunque por las condiciones del mismo deba estar en otro:

```
/*
 * @func: force satellite state using TC
 * @in1: new satellite state
 * @in2: TC ID
 */
void force_satellite_state(unsigned char* new_satellite_state,unsigned short
    tc_id);
```

Para devolver el control de los estados al satélite se puede ejecutar el siguiente código:

```
/*
 * @func: Recover satellite state management
```

```
 * @in: TC ID
 */
void satellite_state_auto(unsigned short tc_id);
```

Se ha implementado un reseteo del sistema de gestión de potencia de dos tipos, mediante software o mediante hardware:

```
/*
* @func: Nanopower soft reset
* @in: TC ID
*/
void eps_softreset_tc(unsigned short tc_id);
```

```
/*
* @func:Nanopower hard reset
* @in1:TC ID
*/
void eps_hardreset_tc(unsigned short tc_id);
```

El Nanopower tiene una configuración por defecto para gestionar variables internas, tales como el modo de funcionamiento de los calentadores de las baterías, los convertidores, el modo de carga de las baterías mediante los paneles solares, el retraso en el encendido de los subsistemas, etc. Por lo que se ha implementado una función que recibe la configuración actual y otra que puede modificarla desde tierra.

```
/*
* @func: set EPS configuration
* @in1: pointer to configuration structure
* @in2: TC data length
* @in3: TC ID
*/
void cepheus_eps_config_set(unsigned char* config_tc, unsigned short tc_length,
    unsigned short tc_id);
```

```
/*
* @func: receive EPS configuration
* @in1: TC ID
*/
void cepheus_eps_config_get(unsigned short tc_id);
```

La configuración que presenta por defecto la plataforma Nanopower es la siguiente:

```
--------------------------------------------------------------------------------
Mode of PPT tracker[1 = AUTO, 2 = FIXED]: 2
Batery heater [0 = Manual, 1 = Auto]: 0
Batery heater low [degC]: 0
Batery heater high [degC]: 5
Nominal mode output value: 0, 0, 0, 0, 0, 0, 0, 0
Safe mode output value: 0, 0, 0, 0, 0, 0, 0, 0
Output switches: init with these on delays [s]: 0, 0, 0, 0, 0, 0, 0, 0
Output switches: init with these off delays [s]: 0, 0, 0, 0, 0, 0, 0, 0
Voltage of boost converters: 3700, 3700, 3700 mV
--------------------------------------------------------------------------------
```

6.6 Gestión de la memoria no-volátil

Para la gestión de la memoria no-volátil se han definido una serie de variables como son la dirección de la memoria, el tamaño de los bloques y los valores de las cabeceras. Además se ha definido la estructura para almacenar los datos, consistente en un identificador seguido de la longitud de los datos a guardar y el checksum de los datos guardados.

```
#define FLASH_BASE 0x48000000  //beginning of flash memory
#define FLASH_16_MAXBLOCK 63 //number of blocks
#define FILE_HEADER 0x55  //header for data
#define BLOCK_16_SIZE 0x10000 //block size

/*
* Data struct for data in flash memory
*/
typedef union __attribute__ ((__packed__)) _cepheus_flashdata
{
  unsigned char raw[9]; /**< Unformatted Time TC */
  struct __attribute__ ((__packed__))
  {
    unsigned char config_id;
    unsigned int data_length; //in bytes
    unsigned int checksum;
  } fields; /**< Struct with individual fields*/
} cepheus_flashdata;
```

Con la siguiente función se realiza el chequeo de los datos guardados en la memoria y borrado de los sectores que presenten datos corruptos.

```
/*
 * @func: check number of experiments saved in flash memory
 * @out: number of experiment saved
 */
unsigned char flash_exp_check(void)
```

Para la lectura de los datos es necesario introducir en la siguiente función el puntero donde se quiere almacenar los datos extraídos, seguido del número del experimento que se quiere recuperar.

```
/*
 * @func: read experiment saved in flash
 * @out: error code
 * @in1: pointer for data extraction
 * @in2: number of the experiment
 */
unsigned char read_exp_flash(char* exp_ram, int exp_numb)
```

Para escribir en la flash se introduce el puntero de los datos a guardar junto con la longitud de los mismos.

```
/*
 * @func: write experiment in flash (prepare header & search an empty block)
 * @out: error code
 * @in1: pointer to data experiment
 * @in2: data length
 */
unsigned char write_exp_flash(char* pexperiment, unsigned int exp_length)
```

La comprobación de archivos se realiza mediante la siguiente función, teniendo como parámetros de entrada el puntero a los datos y el número del bloque de memoria a comprobar.

```
/*
 * @func: check pointer & test for data corruption
 * @out: error code
 * @in1: pointer to data header
 * @in2: block number
 */
unsigned char file_check(char* pdata,int block_number)
```

El borrado de un bloque de memoria se realiza con el siguiente código:

```
/*
 * @func: erase experiment saved in flash memory
 * @in: number of the experiment saved
 * @out: error code
 */
unsigned char erase_exp_flash(int exp_numb);
```

6.7 Gestión del sistema ADCS

Al igual que otros subsistemas, la gestión de la actitud del satélite presenta una función para su inicialización.

```
/*
* @func: ADCS system initialization
*/
void adcs_init(void);
```

Y el propio hilo de la tarea de supervisión y control de los sensores y actuadores.

```
/*
* @func: ADCS supervisor task
* @in: pointer to a parameter (not used). Refer to freeRTOS Manual
*/
void vTaskADCS(void* parameters);
```

7 Test y pruebas

Una parte fundamental del proceso de implementación de software espacial se encuentra en la fase de tests y pruebas. Todos los algoritmos desarrollados durante este proceso deben ser validados en tierra antes de la puesta en órbita de la plataforma. En muchos casos esta fase es incluso más extensa que la del desarrollo del software, ya que se debe intentar reproducir una gran cantidad de situaciones y fallos que pudieran ocurrir durante la vida operativa del satélite.

Sobre estas pruebas se irán implementando versiones de código que corrijan los errores detectados o mejoren el rendimiento ante las situaciones de estudio.

7.0.1 Montaje para ensayo

Para la realización de las pruebas se reproducen las condiciones, desde el punto de vista de comunicaciones, de la operación de la plataforma en el entorno espacial. Por lo tanto es necesario modelar la estación terrena, tanto en software como en hardware. Se utiliza un ordenador con el código de la estación terrena cargado y se hace uso de un VHF-UHF Transceiver conectado mediante usb, Figura 7.1.

Para evitar posibles daños al satélite, este se encontrará en todo momento dentro de su maletín portátil y estará equipado con unos atenuadores en sus antenas. Aparte, se conectará un polímetro para monitorizar el estado de la carga de las baterías.

Para el correcto funcionamiento de la plataforma, es necesario el uso de una fuente de alimentación conectada al EPS-EGSE, si no se va a realizar una prueba con las baterías. Además esta placa debe estar provista

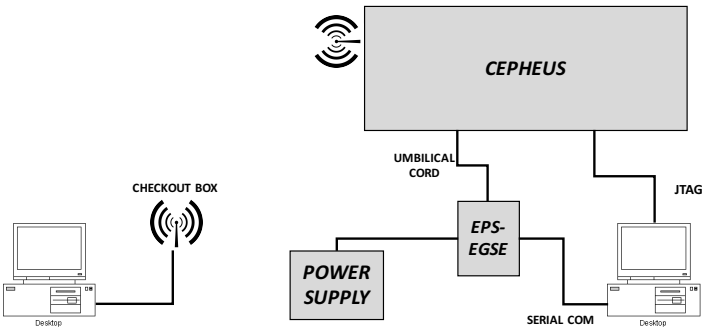

Figura 7.1 Esquema del montaje.

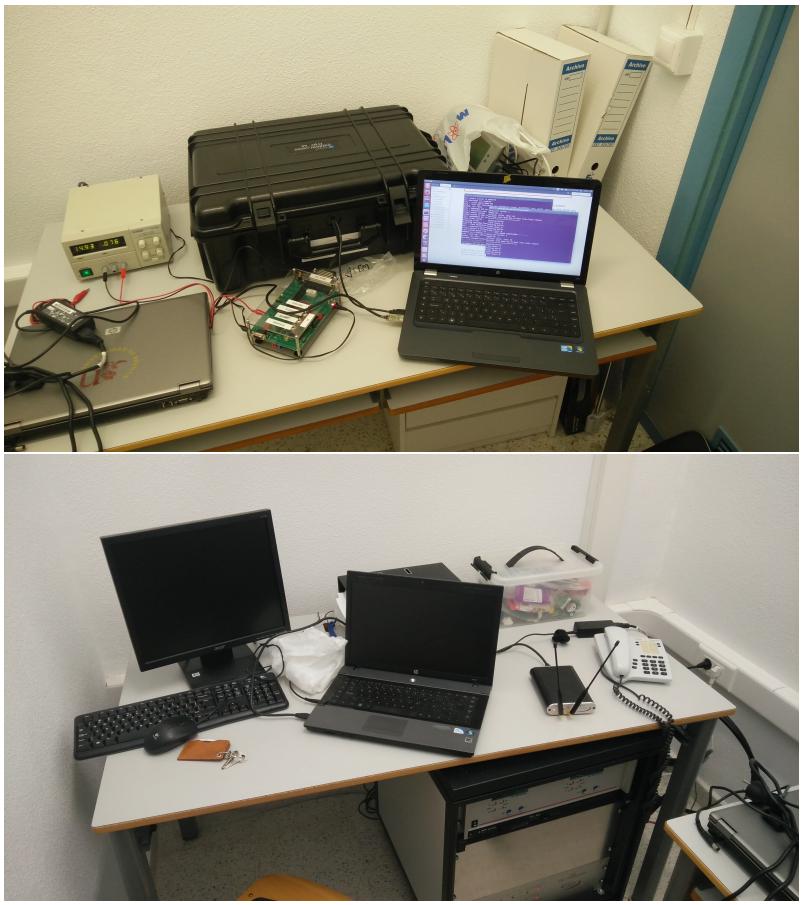

Figura 7.2 Montaje para prueba de integración.

de alimentación, ya que será la encargada de configurar el comportamiento del satélite; el funcionamiento o no de las baterías, el modelado del comissioning y la habilitación de los buses de 3V3 y 5V.

Para la puesta a punto del montaje se han de seguir las instrucciones contenidas en el Anexo II de este trabajo. En la Figura 7.10 puede observarse el ordenador con el VHF-UHF Transceiver y los equipos necesarios para el encendido del satélite.

7.1 Otros proyectos en funcionamiento

Durante la realización de este proyecto, se acordó la cooperación entre la Universidad de Luleå y de Sevilla. La Universidad de Sevilla ha puesto a disposición la estación terrena para su uso en otras misiones de cubesat de bajo coste. Se puede observar la recepción de housekeeping del satélite QB50p1:

```
################################################################################
Decoded Beacon
--------------
================================================================================
                          QB50P Beacon Information
--------------------------------------------------------------------------------
--------------------BEACONFRAME FIELD INFORMATION---------------------------
||                       Source callsign|                        QB50P1||
||                  Destination callsign|                        QB50P1||
||                           Software ID|                    V2 software||
||                          Satellite ID|                        QB50P1||
||                            Frame Type|                 AX.25 beacon #1//
||                       Operational Mode|                   Nominal mode||
```

```
||                        Boot Counter|                          599||
||                      Packet Counter|                          991||
||                   Commands Received|                            0||
||                      Commands Valid|                            0||
||                    Satellite Uptime|                         3995||
||                        Data Valid 1|                         0xff||
||                        Data Valid 2|                         0xff||
||                        Data Valid 3|                         0x3f||


----------------------TRXUV FIELD INFORMATION--------------------------------
||                         TRXUV Doppler|                          152||
||                           TRXUV RSSI|                          170||
||                  TRXUV Reflected power|                  14.2291 mW||
||                   TRXUV Forward power|                  12.7533 mW||
||                      TRXUV TX Current|                    24.49 mA||
||                      TRXUV RX Current|                    93.22 mA||
||                   TRXUV PA Temperature|               149.462 deg. C||
||                      TRXUV Bus Voltage|                   4.32257 V||


----------------------ANTENNA FIELD INFORMATION------------------------------
||          Antenna deployment status - A|                          0x00||
||               Antenna temperature - A|               152.372 deg. C||
||          Antenna deployment status - B|                          0x00||
||               Antenna temperature - B|                152.08 deg. C||


-------------ELECTRICAL POWER SUBSYSTEM(EPS) FIELD INFORMATION---------------
||              Boost Converter 1 Voltage|                       359 mV||
||              Boost Converter 2 Voltage|                       508 mV||
||              Boost Converter 3 Voltage|                       391 mV||
||                       Battery Voltage|                       561 mV||
||              Boost Converter 1 Current|                        30 mA||
||              Boost Converter 2 Current|                       193 mA||
||              Boost Converter 3 Current|                       221 mA||
||             Total photovoltaic current|                       100 mA||
||                  Total system current|                        64 mA||
||      Switched channel current - 3v3 #1|                      6 mA||
||      Switched channel current - 3v3 #2|                      4 mA||
||      Switched channel current - 3v3 #3|                      3 mA||
||      Switched channel current - 5v #1|                      75 mA||
||      Switched channel current - 5v #2|                      1 mA||
||      Switched channel current - 5v #3|                      0 mA||
||          Boost Converter 1 Temperature|                    4 deg. C||
||          Boost Converter 2 Temperature|                    4 deg. C||
||          Boost Converter 3 Temperature|                    3 deg. C||
||                   Battery Temperature|                    1 deg. C||
||                        Channel status|                        0x9||
||                        EPS boot cause|                        0x4||
||                          Battery mode|                     Normal||
||           EPS Powerpoint tracking mode| Maximum Power Point Tracking||


-------------------SOLAR PANELS FIELD INFORMATION---------------------------
||              Solar panel 0 temperature|               21.4844 deg. C||
||              Solar panel 1 temperature|               17.9844 deg. C||
```

Tabla 7.1 Pruebas del sistema de gestión de potencia.

PRUEBA REALIZADA	COMPORTAMIENTO	VERIFICADO
Estabilidad del hilo	El hilo es estable en el sistema operativo y no se produce reboot del OBDH	X
Inicialización gestión de la potencia	El sistema EPS se inicializa correctamente mediante el bus I2C	X
Modo safe por temperatura	El sistema pasa a modo seguro cuando se simula una T fuera de rango. Bastante estable y sin oscilaciones gracias a la histéresis	X
Modo safe por V	El sistema cambia a modo safe cuando el voltaje disminuye por debajo del valor prefijado. Estable y sin oscilaciones	X
Denegación de peticiones por safe mode	Interacción correcta con el sistema de comunicaciones para informar del cambio de estado	X
Modo safe por fallo RTC	Prueba de inicialización sin el reloj rtc reportando error y pasando al modo safe	X

```
||              Solar panel 2 temperature|              3.48438 deg. C||
||              Solar panel 3 temperature|              14.9844 deg. C||
||              Solar panel 4 temperature|              25.4844 deg. C||


--------------------V2 OPERATIONS FIELD INFORMATION--------------------------
||                    SU last response ID|                        0x0||
||            SU thermocouple temperature|                -273 deg. C||
||                        Log OK markers|                       0x77||
||                       WOD log entries|                       3880||
||                        SU log entries|                       3880||
```

Este sistema de housekeeping se implementó también en la plataforma CEPHEUS, ya que se utiliza la señal beacon para enviar el estado del satélite en vez de un paquete de datos sin información.

7.2 Pruebas realizadas para la validación del código

En este apartado se presentan las diferentes pruebas realizadas para validar el software embarcado, teniendo en cuenta que estas deben ser entendidas como las pruebas básicas a nivel de módulo funcional, es decir, serán necesarias pruebas que involucren al satélite y a las distintas situaciones que pueden presentarse. En este proyecto se ha realizado de forma satisfactoria una prueba funcional de la integración a nivel de satélite del software desarrollado.

7.2.1 Gestión de la potencia

Las tareas realizadas en el sistema de gestión de la potencia vienen recogidas en la Tabla 7.1. En la Figura 7.3 puede observarse un HK del subsistema de potencia de CEPHEUS.

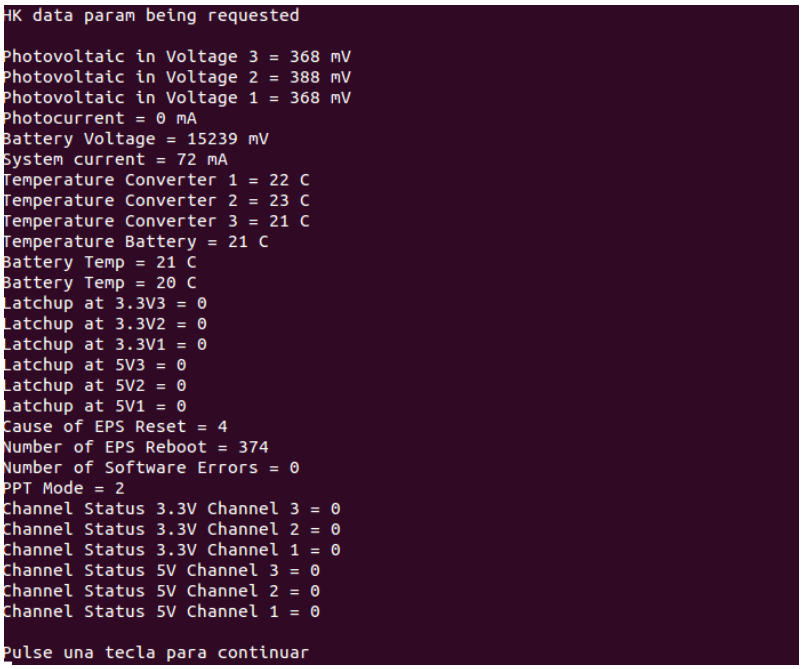

Figura 7.3 Captura de pantalla del HK del sistema EPS.

7.2.2 Gestión de las comunicaciones

En el sistema de gestión de las comunicaciones se han realizado diferentes pruebas según el modo de funcionamiento; beacon, connection o TXRX. En la Tabla 7.2 se reflejan las pruebas para el modo beacon. En 7.3 se tienen las pruebas para connection cuando se dispone del permiso por parte del OBDH. Para la Tabla 7.4 se recogen las pruebas cuando no se dispone del permiso del OBDH y por último, en la Tabla 7.5 se tienen las realizadas para la transmisión y recepción.

7.2.3 Gestión de errores

Las pruebas verificadas para el sistema de gestión de errores vienen recogidas en la Tabla 7.6.

7.2.4 Gestión de los telecomandos

Las verificaciones realizadas a este sistema vienen reflejadas en la Tabla 7.7.

7.2.5 Gestión de memoria no-volátil

En la Tabla 7.8 se tienen las diferentes pruebas realizadas al algoritmo de control de esta subtarea. En la Figura 7.4, se tiene una captura de pantalla de una prueba del código, en la que se extraen los datos y se comprueba el checksum.

7.2.6 Gestión del sistema ADCS

El sistema de gestión ADCS ha sido sometido a las verificaciones de la Tabla 7.9.

Tabla 7.9 Pruebas sistema ADCS.

PRUEBAS REALIZADAS	COMPORTAMIENTO	VERIFICACIÓN
Estabilidad del hilo	Hilo estables sin reinicios inesperados	X
Inicialización del sistema	Inicialización correcta del subsistema mediante el bus I2C	X

Tabla 7.2 Pruebas de comunicaciones en modo beacon/wait [3].

PRUEBA REALIZADA	COMPORTAMIENTO	VERIFICACIÓN
El subsistema de comunicación opera en el estado Deployment/Survival	La radio permanece encendida en modo Beacon con T = 120 segundos	X
El subsistema de comunicación funciona en el estado Normal con hora de conexión	Si el temporizador de conexión ha expirado, la radio se encuentra en modo Beacon con T = 5 segundos. Si no ha expirado la radio se encuentra apagada esperando a que expire	X
Llega petición de conexión en tiempo esperado (modo normal)	Se pasa al estado CONNECTION	X
No llega petición de conexión en tiempo esperado (modo normal)	Expira el temporizador Beacon. Se avisa periódicamente al OBDH de que el satélite se ha perdido y se entra en estado WAIT	X
El subsistema de comunicación funciona en el estado Normal sin hora de conexión	Se avisa periódicamente al OBDH de que el satélite está perdido para que entre en modo Supervivencia	X
El subsistema de comunicación pasa del estado Normal al estado Supervivencia	Se para el temporizador Beacon y se cambia el periodo de Beacon de 5 a 120 segundos	X
El subsistema de comunicación pasa del estado Supervivencia al estado Normal con hora de conexión válida	Se apaga la radio y se activa el temporizador de conexión	X
El subsistema de comunicación pasa del estado Supervivencia al estado Normal sin hora de conexión, con hora de conexión pasada o con hora de conexión errónea	Se avisa periódicamente al OBDH de que el satélite está perdido para que entre en modo Supervivencia	X

Tabla 7.3 Prueba comunicaciones modo connection con permiso del OBDH [3].

PRUEBAS REALIZADAS	COMPORTAMIENTO	VERIFICACIÓN
El segmento terreno envía un TC de conexión	El satélite envía confirmación y pasa al estado TXRX a esperar el siguiente TC	X
El segmento terreno envía un TC distinto a conexión o no encriptado	El satélite envía una trama de error y vuelve al estado BEACON	X

7.2.7 Integración en la plataforma

Una vez verificado el software por separado, comienza la integración de todas las tareas en la plataforma. Se han realizado las siguientes pruebas de verificación, recogidas en la Tabla 7.10. Además en la Figura 7.10 se observa el montaje de la plataforma para la prueba de integración.

El proceso de integración se ha realizado junto con el autor del Trabajo Fin de Grado [3], el cual ha realizado el desarrollo del subsistema de comunicaciones. Estas pruebas consistieron en una comprobación de la estabilidad general del satélite. Se procedió a establecer comunicación y chequear que se modificaban

Tabla 7.4 Pruebas comunicación en modo connection sin permiso OBDH [3].

PRUEBAS REALIZADAS	COMPORTAMIENTO	VERIFICACIÓN
El segmento terreno envía un TC de conexión	El satélite responde que no tiene permiso del OBDH para establecer comunicación y queda en el estado CONNECTION hasta que expira el temporizador de desconexión o recibe otro TC	X
El segmento terreno envía un TC de actualizar hora de conexión	El satélite actualiza la próxima hora de conexión y envía respuesta del segmento terreno	X
Expira el temporizador de desconexión	El satélite vuelve al estado BEACON	X
Llega TC de forzar conexión	El satélite envía confirmación y pasa al estado TXRX a esperar el siguiente TC	X
El segmento terreno envía un TC distinto a conexión o no encriptado	El satélite envía una trama de error y vuelve al estado BEACON	X

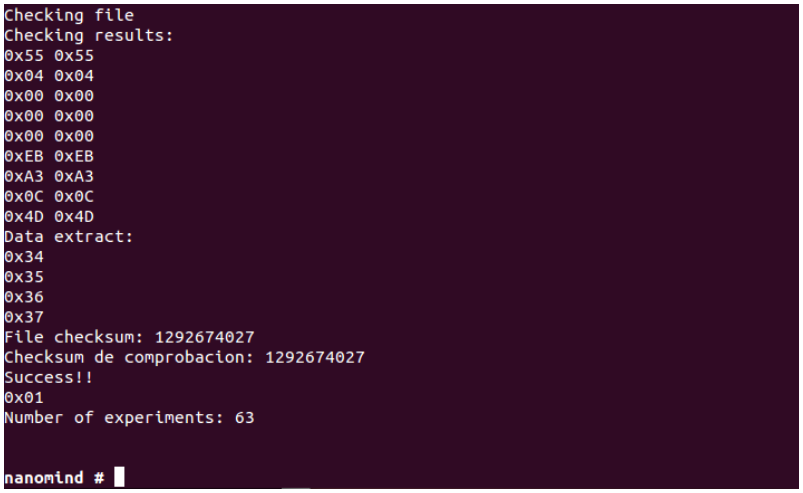

Figura 7.4 Captura de pantalla de una prueba de memoria no-volátil.

los estados del sistema de comunicaciones. Se llevó a cabo un reseteo y un análisis del HK en modo beacon, todo con resultados satisfactorios.

Tabla 7.5 Pruebas sistema de comunicaciones modo TXRX citeedutfg.

PRUEBAS REALIZADAS	COMPORTAMIENTO	VERIFICACIÓN
El segmento terreno envía TC de descarga de experimentos (con datos en el buffer)	Comienza la transmisión de datos de experimentos, se envían y confirman paquetes. Una vez confirmado el último paquete el satélite espera al próximo TC o a que expire el temporizador de desconexión. En el segmento terreno se vuelca la información en un fichero de texto, se imprime por pantalla y se envía el siguiente TC en la lista	X
El segmento terreno envía TC de descarga de experimentos y se interrumpe la comunicación (con datos en el buffer)	Comienza la transmisión de datos de experimentos, se envían y confirman paquetes hasta que dejan de recibirse confirmaciones. Una vez reenviado un paquete varias veces se interrumpe el envío del dato y se espera al próximo TC o a que expire el temporizador de desconexión	X
El segmento terreno envía TC de descarga de housekeeping (con datos en el buffer)	Análogo a los datos de experimentos pero con el housekeeping. Además en el segmento terreno se decodifican las variables de estado del satélite	X
El segmento terreno envía TC de descarga de experimentos/housekeeping (sin datos en el buffer)	El satélite responde que no hay el tipo de datos indicado en el buffer del satélite	X
El segmento terreno envía TC para mostrar lista de buffers del satélite	El satélite envía una lista de los datos que hay almacenados en memoria indicando su tipo e identificador y en el segmento terreno se imprime por pantalla	X
El segmento terreno envía TC para borrar alguno de los buffers del satélite (con datos en el buffer)	El satélite borra el buffer indicado de su memoria y envía respuesta al segmento terreno	X
El segmento terreno envía TC para borrar alguno de los buffers del satélite (sin datos en el buffer)	El satélite responde que no hay el tipo de datos indicado en el buffer del satélite	X
Expira el temporizador de desconexión en el satélite	El satélite pasa al estado de desconexión	X

Tabla 7.6 Pruebas del sistema de gestión de errores.

PRUEBAS REALIZADAS	COMPORTAMIENTO	VERIFICACION
Estabilidad del hilo	Hilo estable sin reboot del sistema OBDH	X
Inicialización del chequeo al reinicio de los subsistemas	Se comprueba la correcta inicialización de los sistemas y el reporte de los errores encontrados cuando se ha simulado un error en un subsistema	X
Inicialización de la gestión de errores	Inicialización realizada correctamente, creación de las colas e inicialización de variables	X
Reporte de los errores	Errores reportados al subsistema de comunicaciones satisfactoriamente	X

Tabla 7.7 Pruebas de la gestión de los telecomandos.

PRUEBAS REALIZADAS	COMPORTAMIENTO	VERIFICACIÓN
Estabilidad del hilo	Hilo estable sin reboot del sistema	X
Inicialización del sistema de gestión de telecomandos	Se inicializan de forma correcta el sistema de gestión de telecomandos, tanto las colas como las variables	X
Cambio del RTC de la plataforma	Se realiza el cambio de hora en el RTC de manera satisfactoria, devolviendo un,mensaje de confirmación	X
Soft reset OBDH	Se realiza el reseteo del OBDH correctamente, inicializándose el sistema sin ningún error	X
Petición de HK del sistema EPS	Se recibe el HK del EPS de forma correcta	X
Forzar estado del satélite	El satélite cambia de estado y devuelve un mensaje con la hora en la que se ejecutó el telecomando	X
Devolver el control del estado del satélite	Se reinicia el hilo de gestión de potencia para automatizar el estado de la plataforma	X
Forzar soft reset desde el sistema EPS	Reinicio de la plataforma	X
Forzar hard reset desde el sistema EPS	Reinicio de la plataforma	X
Devolver la configuración del EPS	Se recibe la configuración del EPS	X
Cambiar configuración del sistema EPS	Se recibe mensaje de confirmación para el cambio de EPS (este vuelve al estado inicial si se produce un apagado)	X

Tabla 7.8 Pruebas sistema de gestión de memoria no-volátil.

PRUEBAS REALIZADAS	COMPORTAMIENTO	VERIFICACIÓN
Lectura de la memoria	Se recibe en la dirección del puntero la información. Si el checksum no coincide se borra el sector	X
Escritura en la memoria	Se realiza satisfactoriamente la escritura de los datos y el cálculo del checksum	X
Chequeo inicial de memoria	Se realiza la lectura de todos los bloques de memoria, borrándose aquellos que presenten discrepancias en los checksums	X

Tabla 7.10 Verificación de la integración de la plataforma.

PRUEBAS REALIZADAS	COMPORTAMIENTO	VERIFICACIÓN
Comprobación de la estabilidad del sistema	No se producen reinicios inesperados del sistema. Se ejecutan correctamente tareas de reinicio remoto y HK sin errores	X
Cambio del estado del sistema de comuniaciones	El sistema de comuniaciones entra en modo conexión cuando se establece contacto con la estación terrena. Si se pierde el mismo, entra en modo beacon satisfactoriamente	X
Solicitud de HK	Se recibe el HK de forma correcta, compresión de datos y encriptación sin errores	X
Solicitud de reinicio	Se reinicia remotamente de forma satisfactoria	X
Solictud de cambio de hora	Se procede al cambio de hora del RTC, completándose de forma satisfactoria	X

En el encendido de la plataforma, si no se produce ningún error debe presentarse los siguientes datos:

```
*** CEPHEUS CUBESAT INITIALIZATION ***
***           Software Version: 3.7              ***
************************************************
Loading Kernel...
Generating tasks...
Error management initialization...
Initializing I2C bus...
Initializing EPS bus...
Initializing Solar Panel bus...
Initializing default satellite status...
Launching Task Power...
Creating Queues between OBDH and COM subsystem...
Setting semaphores...
Launching Task Error...
Starting boot report...
Testing I2C bus...
Starting boot report...
Testing EPS system connection...
Starting boot report...
Testing TRXUV system connection...
Starting boot report...
Testing ANT system connection...
Starting boot report...
Testing ADCS system connection...

-------TASK POWER SUPERVISOR----------
SATELLITE STATUS: 1
ERROR HK: -1
Voltaje: 15557 Temp: 28
****************************
```

8 Conclusiones y trabajo futuro

Para el desarrollo del presente trabajo se ha partido desde una plataforma ya construida y completamente ensamblada a nivel de hardware. En cambio, el software se encontraba repartido entre distintas bibliotecas de los suministradores de los subistemas, las cuales no estaban relacionadas entre sí y en muchos casos no eran operativas. Es por esto por lo que el proyecto, además del diseño e implementación del sistema OBDH, ha consistido en la puesta a punto de la plataforma CEPHEUS en cuanto a software se refiere.

Se ha realizado gran cantidad de trabajo en resolución de problemas, unificación de las bibliotecas, puesta a punto de la estación terrena, etc. Ha sido necesario realizar un estudio por problemas eléctricos con el bus de datos y las cargas de pago. Todo ello ha contribuido a que el resultado final en cuanto al alcance y estabilidad del sistema final no sea el de una plataforma plenamente operativa.

Este proyecto debe entenderse como la realización de la base fundamental sobre la cual descansarán los posteriores desarrollos y versiones del software del satélite. Se ha llegado a conseguir una plataforma relativamente estable, a partir de la cual se pueden ir añadiendo funcionalidades. Por lo tanto se ha dado un salto cualitativo en cuanto al desarrollo de la plataforma, dejando las mejoras para posteriores trabajos o tesis, que irán poco a poco evolucionando el producto hasta llegar al resultado final.

En cuanto a las pruebas y test, los realizados en el proyecto han sido satisfactorios y han servido para garantizar que las funciones implementadas cumplen con sus objetivos. Pero esto no significa que no deben realizarse más pruebas para garantizar la estabilidad del sistema ante cualquier situación comprometida o imprevista. La filosofía de diseño KISS ha sentado las bases para un sistema operativo que sea difícil de bloquear. Además, el hecho de separar las cargas de pago mediante una interfaz física hace que la gestión de los experimentos no sobrecargue el tiempo de procesamiento del OBDH y evite situaciones de riesgo. La plataforma nominal del cubesat se concentrará en las funciones vitales del satélite y los experimentos serán gestionados con su propio procesador, siempre y cuando se tenga autorización por parte del OBDH. Con todo esto además se consigue una plataforma totalmente independiente de las cargas de pago que se implementen, pudiendo mantener el diseño del software y dejando a cargo del cliente el desarrollo del suyo, cumpliendo siempre las especificaciones y requisitos que se le proporcionen en cuanto a seguridad y consumo de recursos.

8.1 Líneas de trabajo futuras

Como se ha comentado antes, partiendo de esta base presentada en el trabajo, pueden mejorarse aspectos de la plataforma. Además de funcionalidades para futuros subsistemas que se implementen hay ciertas ramas que serían de gran utilidad de cara a futuros trabajos.

8.1.1 Sistema de gestión de potencia

Dentro del sistema de gestión de la potencia, una línea interesante de investigación se encuentra en la mejora de la estimación de la energía disponible en las baterías. En este proyecto se ha optado por la estimación de la carga según el voltaje de salida, lo que no es una medida del todo exacta y menos en las baterías de litio. Estas tienen una curva de descarga bastante planas y por lo tanto dentro de un pequeño rango de voltaje puede existir una gran cantidad de estados de carga, Figura 8.1.

Una forma de mejorar este proceso puede consistir en la implementación de un algoritmo de estimación del estado de carga (SoC) de las baterías teniendo en consideración factores como la temperatura o el State of Health(SoH).

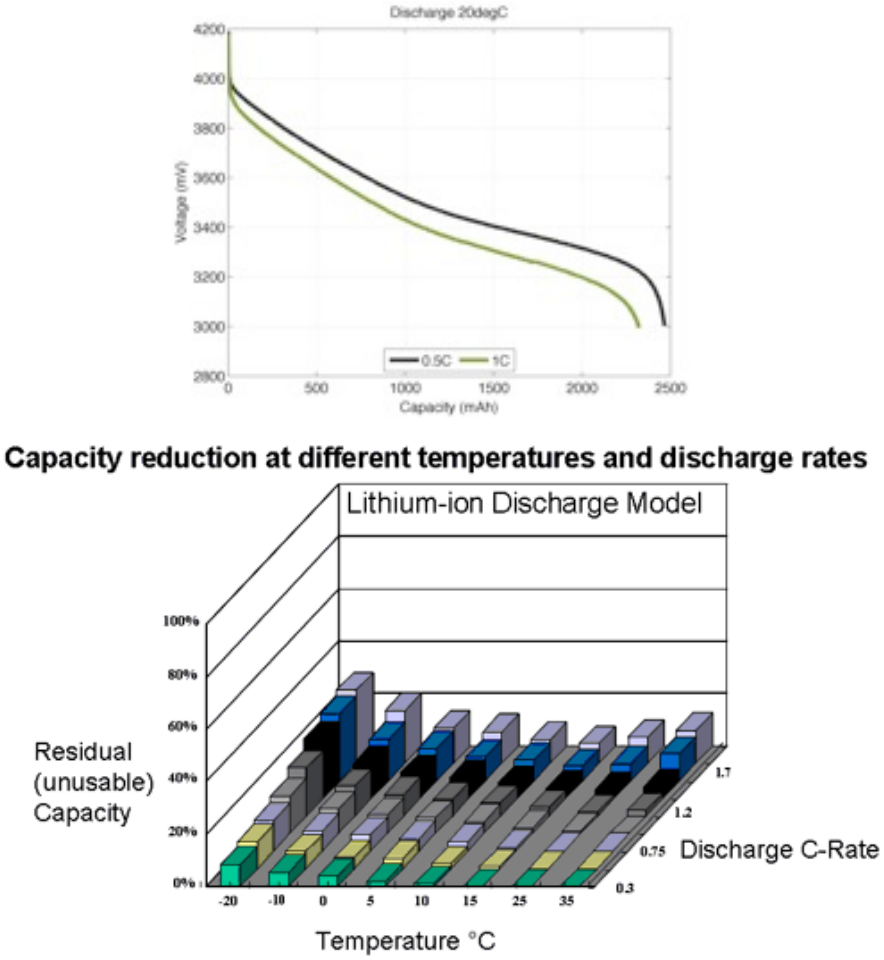

Figura 8.1 Comportamiento de las baterías de ion-litio.

8.1.2 Sistema ADCS

En el sistema de control de actitud es una de las ramas en las que más se puede avanzar. Se propone como trabajo futuro la elaboración de un sistema de detumbling con un control B-dot, según el cual el par ejercido por los actuadores es calculado mediante la primera derivada del campo magnético. Para su correcta validación es necesario la elaboración de un modelo matemático de la dinámica del satélite y su interacción con las fuerzas ejercidas por los magneto torques cuando existe un campo magnético. Así mismo se tendría que modelar la adquisición de los datos por parte de los sensores magnéticos del Nanomind.

8.1.3 Sistema de gestión de la memoria no volátil

Para la gestión de la memoria se tiene como línea futura la investigación para implementar un sistema de archivos dentro de la plataforma, que sea más eficiente con los recursos de los que dispone la memoria flash. Además la detección y gestión de errores puede ser más precisa, intentando que cuando se produzca un fallo permanente en la batería no se inutilice un sector entero, sino la dirección de memoria afectada.

Puede también ser conveniente desarrollar un sistema de trasvase de código desde la memoria flash de código a la RAM, o a la flash de almacenamiento de datos. Con esto se evitaría que una degradación de la memoria no-volátil del código ponga en riesgo la integridad del satélite.

8.1.4 Optimización de la gestión de hilos

La optimización del rendimiento del sistema operativo del OBDH puede ser un tema de investigación bastante interesante. La implementación de algoritmos que gestionen las prioridades de los hilos; evitando bloqueos en el sistema y aumentando la eficiencia de los recursos de procesamiento.

Recalcar que en el presente trabajo se han tomado prioridades de los hilos como un valor constante, dejando a las tareas más críticas con una prioridad mayor, pero siempre teniendo en cuenta que debe existir un tiempo de "descanso". Si un hilo no depende de los demás y tiene la prioridad más alta, ocupará todo el tiempo de procesamiento de la plataforma y terminará por provocar un soft-reset debido a que no se ha refrescado el WDT durante este proceso.

Apéndice A
Herramientas para la gestión de la documentación

A.1 Plantilla Excel MOC

Matriz de conformidad de los requerimientos generales de la misión Cepheus							
Código del Requerimiento	Descripción	Fecha incorporación	Cumplimiento	Comentarios			
REQ-OB-N01	El satélite tendrá la capacidad de recibir y llevar a cabo un comando de apagado.	26/2/16			NºReq	Comple	% Completado
REQ-OB-N02	El tiempo de ejecución de la orden de apagado o encendido de los subsistemas se encontrará acotado.	26/2/16			58	0	0
REQ-OB-N03	Todos los componentes desplegables como son las antenas , deberán esperar un mínimo de 30 minutos para su despliegue después de que los interruptores de lanzamiento se activen tras la eyección del P - POD.	26/2/16					
REQ-OB-N04	Las 48 horas posteriores a la inserción del satélite en su órbita fuera del lanzador serán reservados para la carga de las baterías y la puesta en marcha de los subsistemas básicos de la plataforma.	26/2/16					

Figura A.1 Plantilla MOC.

A.2 Plantilla Excel para el cumplimiento de las tareas

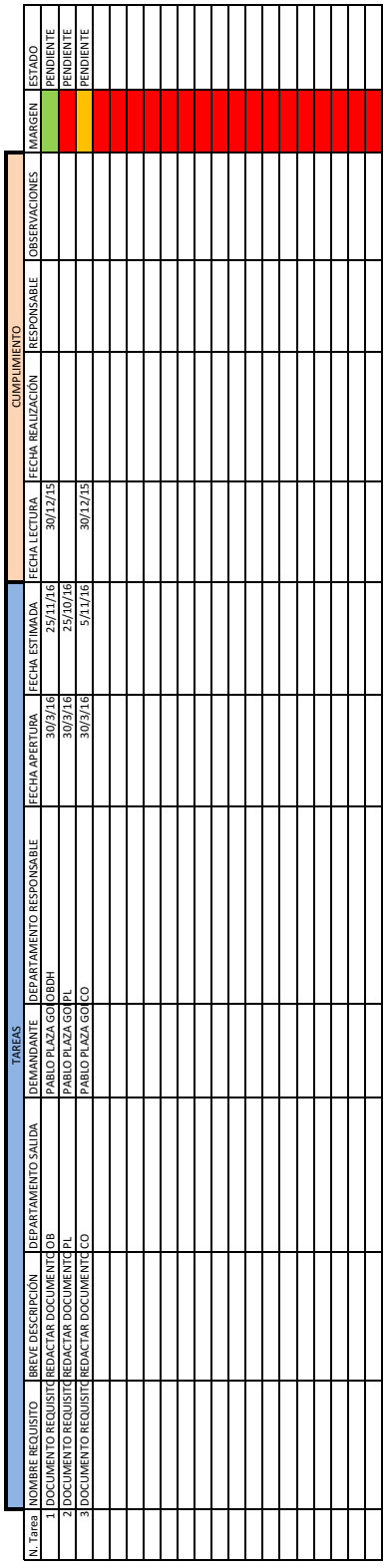

Figura A.2 Plantilla tareas.

A.3 Plantilla Excel Planning

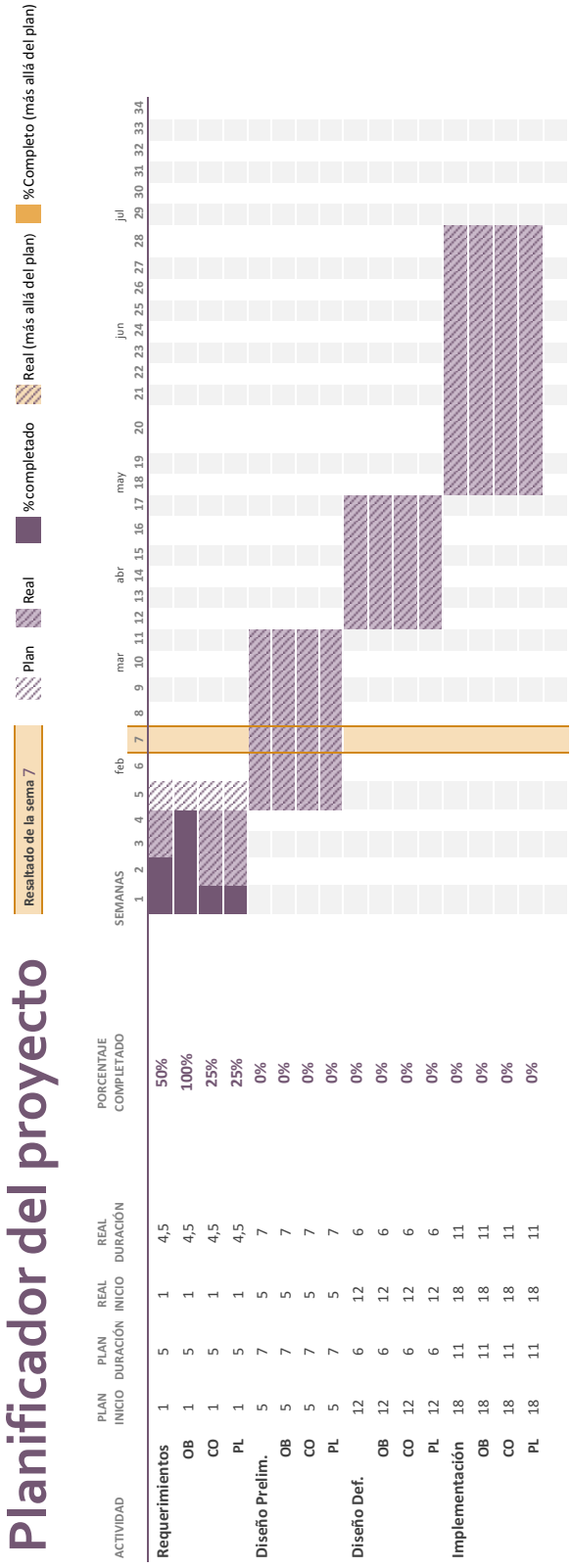

Figura A.3 Plantilla diagrama de Gant.

Índice de Figuras

Índice de Tablas

Bibliografía

[1] Karim Djouani Ajayi Michael Oluwatosin, Yskandar Hamam, *Attitude control of a cubesat in a circular orbit using magnetic actuators*, IEEE (2013).

[2] Richard Barry, *Using the freertos real time kernel*, freeRTOS, 2009.

[3] Eduardo Jesús Luque Calderón, *Sistema de comunicaciones entre el satélite cepheus y la estación terrena*, Master's thesis, Universidad de Sevilla, 2016.

[4] Samir A. Rawashdeh Christopher Mitchell, Jason Rexroat, *Development of a modular command and data handling architecture for the kysat-2 cubesat*, IEEE (2014).

[5] Jens Eickhoff, *A combined data and power management infrastructure*, Springer, 2013.

[6] Gomspace, *Nanopower battery datasheet*, Gomspace, Agosto 2015.

[7] ———, *Nanopower p-series datasheet*, Gomspace, Septiembre 2015.

[8] Bill Jackson, *A robust fault protection architecture for low-cost nanosatellites*, IEEE (2015).

[9] Simon Lee, *Cubesat design specification rev 12*, California Polytechnic University (2005).

[10] James Martin, *Diseño de sistemas de computadores en tiempo real*, Diana, 1981.

[11] Javier Casado Pérez, *Historia y tecnología de la exploración espacial*, Cockpit Studio, 2002.

[12] Linhart Richard, *A three axis magnetometer for use in a small satellite*, Univerzitni Pilsen (2006).

[13] Manuel Rodríguez, *Solarmems-end-en05 gestión del software*, SolarMEMS, 01 ed., Abril 2014.

[14] James E. Lumpp Samuel F. Hishmeh, Tyler J. Doering, *Design of flight software fot the kysat cubesat bus*, IEEEAC (2008).

[15] ECSS Secretariat, *Space project management, project phasing and planning*, ESA-ESTEC, Abril 1996.

[16] Sean M. Horton Shaina Johl, E. Glenn Lightsey, *A reusable command and data handling system for university cubesat missions*, IEEE (2014).

[17] Norman G. Fitz-Coy Sharan A. Asundi, *Design of command, data and telemetry handling system for a distributed computing architecture cubesat*, IEEE (2013).

[18] Soren Vejlgaard Vedstesen Torben Graversen, Michael Kvist Frederiksen, *Attitude control system for aau cubesat*, Aalborg University (2002).

[19] José Manuel Quero y Laura León, *Aicia requerimientos generales de la misión cepheus*, AICIA, 2014.

[20] Hans-Peter Roeser y Volker Liebig, *Onboard computers, onboard software and satellite operations*, Springer, 2011.